Efficient Pitch Detection Techniques for Interactive Music

Patricio de la Cuadra, Aaron Master[*], Craig Sapp
Center for Computer Research in Music and Acoustics, Stanford University
email: {pdelac|asmaster|craig}@ccrma.stanford.edu

Abstract

Several pitch detection algorithms are examined for use in interactive computer-music performance. We define criteria necessary for successful pitch tracking in real-time and survey four tracking techniques: Harmonic Product Spectrum (HPS), Cepstrum-Biased HPS (CBHPS), Maximum Likelihood (ML), and the Weighted Autocorrelation Function (WACF).

1 Introduction

Fundamental frequency detection is a well-researched problem often considered "solved" in the case of recorded monophonic voices and instruments. However, the identification of pitch becomes much more difficult in live concert settings. Wrong notes cannot be fixed in retrospect during a performance — in the recording studio you can always do a second take. Therefore, avoiding pitch-tracking errors in real-time audio is much more important than in non-real time audio where errors can be cleaned-up with more robust secondary processing of the pitch-tracking data. The need to avoid initial pitch-tracking errors in interactive music applications generates the following list of requirements for real-time algorithms:

- ability to function in real time
- minimal output delay (latency)
- accuracy in the presence of noise, and
- sensitivity to musical requirements of the performance.

A pitch tracker designed for non-real time applications will not be successful when applied to interactive music if it cannot satisfy these requirements.

Obviously, pitch-tracking algorithms for interactive music must be able to run in real-time. The real issue is one of computational complexity: How much error checking can be added to the system while still allowing the algorithm to run in real-time? For example, several pitch-tracking algorithms could be run in parallel to reduce the error rate, but this may not always be practical. Errors can also be reduced if frequency transforms are heavily overlapped. But there will always be a limit to the amount of processing or programming complexity that can be focused on reducing first-pass pitch-tracking errors—some processor time must be left over for using the resulting pitch measurements.

Aside from minimizing pitch errors, the latency in a pitch detection algorithm is the next most critical factor to consider. Note that there is a difference between attack latency and pitch identification latency. Once a note is played, it takes the ear at least seven cycles to accurately identify a pitch. Therefore the perception of note onsets and note pitches are not directly related. Guesses about the steady state pitch of a note can be made during an attack, but if high-quality pitch information is needed, then you must wait. Real-time Pitch-to-MIDI implementations require very low latency due to note attack requirements, but extracting pitch for key-finding algorithms that monitor what key the musician is currently playing can tolerate very large latencies.

Algorithms which tolerate noisy audio input are particularly desirable in interactive music. Performance environments can contain noise from many sources, such as cheap soundcards and sound from speakers and other instruments. Other performance conditions and compositional requirements cannot always be anticipated; however, we establish the following general criteria to cover the most common characteristics of an interactive music environment needing pitch tracking:

- frequency resolution of at least semi-tones, including the correct octave.
- timely recognition and quality of instantaneous pitch for possible real-time conversion into symbolic pitch.
- instruments with well-behaved harmonics (such as cello and flute).

No pitch algorithm can possibly cover all requirements and unanticipated conditions in interactive music performance. A computer-musician should be knowledgeable of a variety of techniques and algorithmic parameters when implementing real-time pitch-trackers. This paper presents four techniques deemed suitable for interactive music; however, a large number of algorithms which may work as well as or even better cannot all be covered. Other pitch-tracking algorithms useful to examine further include Doval and Rodet (1991), Brown and Puckette (1992) as well as a classical paper on time-domain pitch extraction by Gold and Rabiner (1969).

[*] supported by the National Science Foundation.

Frequency-domain algorithms are generally more robust than time-domain algorithms. Thus, the first three techniques examined below are based on the Discrete Fourier Transform, although the DFT loses resolution at lower frequencies. The ML algorithm behaves well at low frequencies regardless, since it considers the relative position of partials without the need to zero pad the signal frames. HPS is a more computationally efficient technique, and can therefore afford the expense of zero padding. CBHPS is not as inexpensive, but needs less zero padding due to the enhanced low frequency resolution provided by the frequency-indexed cepstral component. The WACF is an accurate but computationally inexpensive time-domain technique although its resolution is limited by the sampling frequency.

2 Pitch Detection Algorithms

2.1 Harmonic Product Spectrum

The HPS pitch-detection algorithm (Noll 1969) is the simplest method to implement and does well under a wide range of conditions. Its primary drawback is the need to enhance low frequency resolution with zero padding of the signal before transforming so that the spectrum can be interpolated to the nearest semi-tone. This wastes a lot of computational power, because high frequencies are also being unnecessarily interpolated as well. However, this algorithm runs very well in real-time tests with a 200 MHz processor.

The HPS algorithm measures the maximum coincidence for harmonics according to equation (1) for each spectral frame, $X\left(\omega\right)$.

$$Y\left(\omega\right)=\prod_{r=1}^{R}\left|X\left(\omega r\right)\right| \tag{1}$$

$$\hat{Y}=\max_{\omega_i}\left\{Y\left(\omega_i\right)\right\} \tag{2}$$

where R is the number of harmonics to be considered ($R=5$, for example), and frequency ω_i is in the range of possible fundamental frequencies. The resulting periodic correlation array, $Y\left(\omega\right)$, is searched for a maximum value, $\hat{Y}$, as is shown in equation (2).

Octave errors are a common problem in pitch measurements from HPS. Almost always in these error cases, the pitch is detected one octave too high. To correct for this error, post-processing should be done with the following rule: IF the second peak amplitude *below* initially chosen pitch is approximately 1/2 of the chosen pitch AND the ratio of amplitudes is above a threshold (*e.g.,* 0.2 for 5 harmonics), THEN select the lower octave peak as the pitch for the current frame.

Figure (1) demonstrates the HPS algorithm graphically. The function $Y(\omega)$ is shown on the far right with the maximum value, $\hat{Y}$, being the most likely pitch for the analysis frame. Due to noise, frequencies below about 50 Hz should not be searched for a pitch.

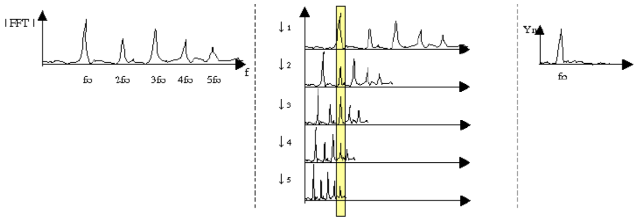

Figure 1: Overview of the HPS algorithm

2.2 Maximum Likelihood

The ML algorithm (Noll 1969) searches though a set of possible ideal spectra and chooses the one which best matches the shape of the input spectrum. The ideal spectrum is defined to be an impulse train starting at frequency ω convolved with the signal window's spectrum.

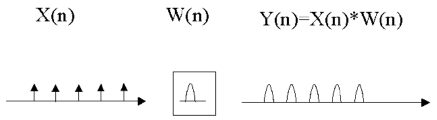

Figure 2: Generation of an ideal spectrum

This process tries to minimize the error between the spectral frame and possible candidate spectra as illustrated in the following equations.

$$\begin{aligned} E(\omega) &= \|Y-\tilde{Y_\omega}\|^2 & (3)\\ &= \|Y\|^2+\|\tilde{Y_\omega}\|^2-2Y\tilde{Y_\omega}^T & (4)\end{aligned}$$

where Y is the given input spectrum, and $\tilde{Y_\omega}$ is an ideal harmonic spectrum with a fundamental frequency located at ω. The term $\|Y\|^2$ remains constant and $\|\tilde{Y_\omega}\|^2$ remains constant for most ω frequencies of interest, so we need to find the maximum value of the product of the two spectra where the most likely pitch $\hat{Y}$ is to be assigned, as shown in equation (5). Figure (3) graphically shows the ML algorithm selection process given in equation (5).

$$\hat{Y}=\min_{\omega}\left\{E(\omega)\right\}=\max_{\omega}\left\{Y\tilde{Y_\omega}^T\right\} \tag{5}$$

Since the ML algorithm does not need to be spectrally interpolated like HPS, smaller transform sizes may be used. This can speed up the algorithm considerably. However, the efficiency of the algorithm depends on how much pitch resolution is required. ML works well when the true pitch is centered on a reference signal in the matrix multiply, but ML

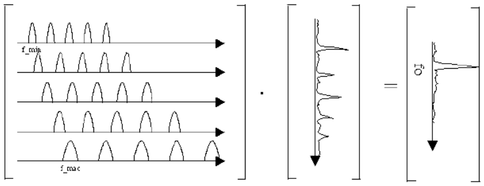

Figure 3: Overview of Maximum Likelihood

gives erratic answers when the input signal is halfway between two cases in the reference matrix; therefore, ML works well if the input source is in a fixed tuning. Keyboard and woodwind instruments are more appropriate for ML than strings or voice since the latter instruments can easily produce non-discrete pitches, particularly in vibrato.

Specifying a limited range of possible pitches can improve the efficiency in ML by lowering the number of test signals in the reference matrix. If the true pitch is outside of the test range by an octave, then ML will detect the correct pitch class in the adjacent reference range. At two octaves outside the ML range, the detection becomes less accurate in determining the pitch class. ML is less tolerant of noise and weak signals than the HPS method.

2.3 Cepstrum-Biased HPS

Possibly the most popular pitch tracking method in speech analysis is the cepstral analysis technique. First, the Cepstrum is calculated by taking the DFT of the log of the magnitude spectrum of a speech frame. Then, the Cepstrum is inspected for a peak in a limited range, corresponding to the period of the signal, as well as for a second or third peak an equal distance (period) away from the first or second peak.

An improved algorithm may be created by combining the Cepstrum with the HPS function (Master 2000). This technique has been used to initialize guesses in a polyphonic pitch detection system. Separately, the Cepstrum and HPS achieve modest success for this task. But combined, they yield fairly reliable results for at least the 2 pitch speech case.

Since these functions exist in different domains, the first step is in combining them is to convert the Cepstrum to be frequency domain indexed. To do this, Cepstrum values at $\textit{Index}_{\text{cep}} = k$ are written to the Frequency Indexed Cepstrum (or FIC) with indices of $\textit{Index}_{\text{FIC}} = \text{floor}(N/k)$ where N is the number of points in the DFT and “floor” is a function specifying the greatest integer less than the argument. The new index represents the frequency bin value corresponding to the time value in the Cepstrum. Thus, a peak tending to indicate a period at value k in the Cepstrum now tends to indicate a pitch with a corresponding value.

The FIC function and the HPS are then multiplied to create a new function, the Cepstrum-Biased HPS (CBHPS). In the CBHPS, spurious peaks in the pitch-doubling-robust FIC tend to be canceled out by corresponding low values in the pitch-halving-robust HPS function, and vice versa. Thus, the peaks seen in the new function have been generally very reliable for the 2 voice speech case. In that case, peaks were chosen as pitch indicators if their frequencies were not multiples of a lower frequency peak, and if their magnitude was at least 15% of the largest peaks’ magnitude. CBHPS is good for application in multi-pitch detection, since it robustly handles noise and pitch errors.

2.4 Weighted Autocorrelation Function

One popular time domain technique is to pick peaks in the autocorrelation function, or ACF. The ACF is created from the equation:

$$\phi(\tau) = \frac{1}{N} \sum_{n=0}^{N-1} x(n)x(n+\tau)$$

and measures the extent to which a signal correlates with a time offset (τ) version of itself. Because a periodic signal will correlate strongly with itself when offset by the fundamental period, we can expect to find a peak in the ACF at the value corresponding to a period.

An alternate to the ACF is the Average Magnitude Difference Function. The AMDF looks not at the product of a signal with a time offset version of itself, but rather at the difference. Thus, the AMDF tends to have valleys where the ACF has peaks. Calculation of the AMDF is less computationally expensive than the ACF due to the lack of multiplication operations (Niesler and Robinson).

The equation for the AMDF is given as:

$$\psi(\tau) = \frac{1}{N} \sum_{n=0}^{N-1} |x(n) - x(n+\tau)|$$

As noted by Kobayashi and Shimamura (1995), these two functions have independent statistics, and may be combined to produce a more noise-robust estimate of pitch, especially in cases where gross pitch error (more than 10 Hz error) is possible. For purposes of single F_0 detection, this method was shown by those authors to be substantially more effective in noisy environments than either the ACF or AMDF alone, or the popular cepstral technique described above. The function described by these authors is:

$$f(\tau) = \frac{\phi(\tau)}{\psi(\tau) + k}$$

where optimal results were achieved with $k = 1$ regardless of SNR.

3 Real-Time Tracking Issues

Pitch tracking in real-time situations usually involves additional steps beyond frame-by-frame pitch detection to enhance the quality of the measured pitch. The pitch detection algorithms defined above generate the instantaneous pitch for the input signal which will invariably contain some tracking errors. In particular, if the input signal changes pitch during an analysis frame or there is any significant transient due to a note attack, the resulting pitch measurement may be misleading. Figure (4) shows the results of HPS pitch tracking a melody performed on a cello. Notice the discontinuities in the pitch detection at transients between notes which must be compensated for in real-time.

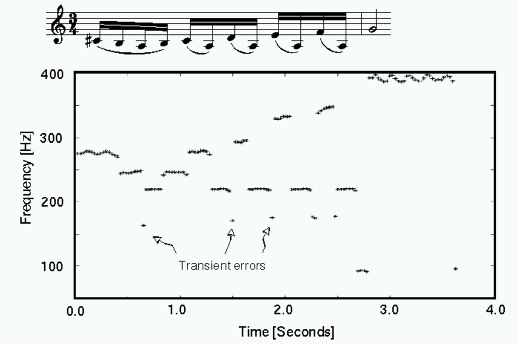

Figure 4: Cello pitch tracking analysis with transient errors

Depending on the application, pitch data can remain continuous or be converted into discrete symbolic form such as MIDI where there are discrete note-on and note-off times for each pitch. For instantaneous pitch control, it is usually sufficient to filter out spikes in the pitch data due to transients and octave errors. It even may not be necessary to remove erroneous pitch depending on the noise tolerances of the application. The pitch algorithm, instrument, and computer setup will determine the required level of error correction measures.

As a model tracking example, figure (5) shows the results of continuous pitch tracking for a slurred chromatic melody (Flight of the Bumble-Bee) performed on a flute. The individual pitches are well separated with no transients, resulting in good data for 12-tone quantization without any secondary error correction necessary.

A good way to prevent accidental pitch assignment to transient regions is to monitor the steady-state behavior and amplitude of the detected pitch over several spectral frames. However, to minimize the latency in the determination of a new note, as few frames as possible must be considered. Since the pitch detection algorithm does not involve re-synthesis of a signal, the signal frame overlap can be done at any arbitrary overlap factor. Correct new pitch updates need to be made within about 30-40 milliseconds to avoid significant perceptual delay. An overlap factor at about 10 milliseconds per frame usually leaves enough time to recover from transients.

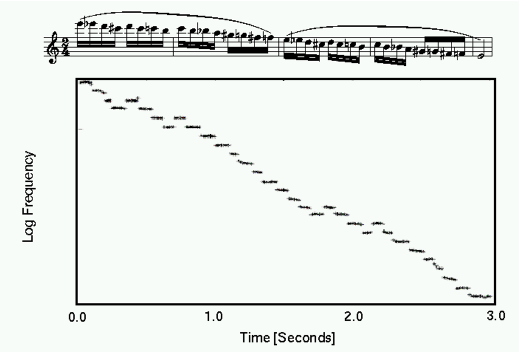

Figure 5: Flute pitch tracking analysis

Musical instruments are not always tuned to a set pitch, and can drift due to temperature and humidity. Some algorithms such as ML are sensitive to tuning. The pitch algorithms should be tuned so that they will not inadvertently oscillate between two tones a minor second apart when only one note is actually being played (hysteresis). This can be compensated somewhat with an initial tuning calibration, or by monitoring the interval size between successive notes to keep the center frequencies of the note well within the range of a twelve-tone quantization of the pitch.

References

Brown, J. and M. Puckette (1992). An efficient algorithm for the calculation of a Constant-Q transform. *Journal of the Acoustical Society of America 92*(5), 2698–2701.

Doval, B. and X. Rodet (1991). Fundamental frequency estimation using a new harmonic matching method. In *Proceedings of the International Computer Music Association*, pp. 555–558. Montreal, Canada.

Gold, B. and L. Rabiner (1969). Parallel processing techniques for estimating pitch periods of speech in the time domain. *Journal of the Acoustical Society of America 46*, 441–448.

Kobayashi, H. and T. Shimamura (Sept. 1995). A weighted autocorrelation method for pitch extraction of noisy speech. In *Proceedings of the Acoustical Society of Japan*, pp. 343–344. Urawa, Japan: Saitama University.

Master, A. (2000). Speech spectrum modeling from multiple sources. Master's thesis, Cambridge University, Engineering Dept., Cambridge, England.

Niesler, T. and T. Robinson. Speech analysis course handouts. Cambridge Univ. Enginnering Dept., 1999.

Noll, M. (1969). Pitch determination of human speech by the harmonic product spectrum, the harmonic sum spectrum, and a maximum likelihood estimate. In *Proceedings of the Symposium on Computer Processing ing Communications*, pp. 779–797. Polytechnic Institute of Brooklyn.