424 Report: Pitch Detection

Becca Wroblewski

I decided to investigate the implementation of a pitch detection algorithm for this project. Pitch detection algorithms have been long established and used in various musical contexts including in tuners and as the first processing scheme in a pitch shifting scheme as in Auto-Tune.

The paper “Efficient Pitch Detection Techniques for Interactive Music” delves into a variety of pitch detection algorithms used in the context of computer-music. The algorithms described include the Harmonic Product Spectrum (HPS), Cepstrum-Biased HPS (CBHPS), Maximum Likelihood (ML), and Weighted Autocorrelation Function (WACF) algorithms. The final approach of the Autocorrelation Function was used in this investigation. The paper presents three methods in the Weighted Autocorrelation Function section. The first is simply the Autocorrelation Function (ACF), expressed below.

The second is the Average Magnitude Difference Function (AMDF), expressed below.

And the third is a method combining the two, expressed in Equation 3.

First, the ACF was implemented in the frequency domain, by applying the approach detailed in the Unbiased Cross-Correlation section of the “Mathematics of the Discrete Fourier Transform (DFT), with Audio Applications”. Autocorrelation is thus performed using the following equations.

The pitch was determined from the ACF then, by applying peak detection performed by the python SciPy find\_peaks function. This function is applied where with the height of the peaks must exceed half of the maximum value of the ACF. The peak with the maximum height is then used to find the lag value corresponding to this peak. The lag value is then used to find the frequency of the signal given where fs is the sampling frequency. This methodology is then applied to the input signal per blocks of length 1024. Finally, the algorithm outputs the note corresponding to the identified pitch value in Hz by applying the librosa library’s hz\_to\_note function. The combined AMDF and ACF methodology was also implemented and tested, however, the AMDF was implemented in the time-domain, yielding very slow processing times for an entire audio signal. The results as applied to the baseline test signals were also slightly worse than the ACF by itself. Thus the ACF+AMDF method was not investigated extensively, however it is of note that noisy conditions were not investigated in this report, where ACF+AMDF may prove more useful.

The ACF methodology explained above was applied to 4 test signals: a 440Hz tone, a flute performing an F Major scale, A violin playing a G#3 note, and a vocalist singing a scale. The following images show the spectrogram of a section of the audio, overlayed with a scatterplot showing the detected pitch (in Hz) for each time step or 1024.

A blue and yellow graph

Description automatically generated with medium confidence A chart of a flute scale

Description automatically generated with medium confidence

A graph of a violin

Description automatically generated with medium confidence A colorful scale with black dots

Description automatically generated with medium confidence

The pitch detection using ACF does a relatively good job at following the pitch. However, there are significant errors especially in accidentally detecting harmonics and in showing random pitches during portions of silence. The algorithm also is worse at detecting pitch at the beginning and ending of a note. The test files chosen also are relatively ideal conditions, showing simpler pitched patterns with relatively low noise and a relatively long duration per pitch.

Further work to improve this algorithm may be done in many ways to improve performance under a wider range of conditions. The paper “YIN, a fundamental frequency estimator for speech and music” performs pitch detection utilizing a difference function somewhat like the AMDF presented in the first paper. The difference equation is shown below.

They then build upon this method by developing the Cumulative Mean Normalized Difference Function shown below.

This function increases the ease of interpreting the difference equation, and, in normalizing the result aids in further methods of refining the pitch detection procedure. Other methods explored in this paper that may be useful in integrating into this pitch detection investigation include Parabolic Interpolation and Best Local Estimate Analysis. Parabolic Interpolation implements an interpolation in the frequency domain to allow for more frequency resolution and would likely improve the precision of the algorithm, especially at lower frequencies. The best local estimate approach also reduces error in pitch detection by essentially running the algorithm twice, once to obtain a first guess, and then again to refine the estimate on a narrower range of possible values.

References

[1] “Pitch Detection Methods,” *sound.eti.pg.gda.pl*. Available: <https://sound.eti.pg.gda.pl/student/eim/synteza/leszczyna/index_ang.htm>

[2] J. O. Smith, *Mathematics of the discrete Fourier transform (DFT) : with audio applications*. Seattle. Wa: Booksurge Publishing (Http://Www.Booksurge.com, 2007.

[3] 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, Apr. 2002, doi: <https://doi.org/10.1121/1.1458024>‌

[4] L. Rabiner, “On the use of autocorrelation analysis for pitch detection,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, no. 1, pp. 24–33, Feb. 1977, doi: https://doi.org/10.1109/tassp.1977.1162905

[5] P. De La Cuadra, A. Master, and C. Sapp, “Efficient Pitch Detection Techniques for Interactive Music.” Available: https://ccrma.stanford.edu/~craig/papers/01/icmc01-pitch.pdf. [Accessed: Mar. 19, 2024]

[6] “Pitch detection using Python and autocorrelation,” *Scicoding*, Dec. 08, 2021. Available: https://scicoding.com/pitchdetection. [Accessed: Mar. 19, 2024]