MIDI Matrix Transcription Using Cepstral and Autocorrelation Based Pitch Detection with Timbre Model Recommendations in MATLAB

Nicholas Bruno[1], Daniel Kannen[1]
[1]Audio Music Engineering, Department of Electrical and Computer Engineering, University of Rochester

Background

Midi has always been a useful protocol to store and play back musical performances. We wanted to create a mechanism for taking already recorded monophonic music and convert it to MIDI.

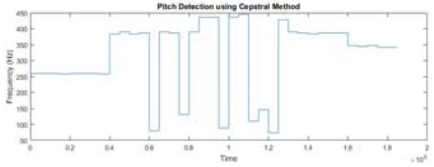
By doing this, we can take musical performances that have already been recorded, and use MIDI and external synthesizers to playback the performance while being able to modify the sound and other variables. We also wanted to integrate timbre modeling so that the data retains some of the information from the original signal besides pitch and length.

Auto-Correlation vs. Cepstral Pitch Tracking

The real cepstrum of a signal s[n] = e[n] * Θ[n] is defined as:

$$c[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} log|S(w)|e^{jnw}dw$$

Cepstrum is a Fourier analysis of the logarithmic amplitude spectrum of the input signal.[1] Initially experimenting with Cepstral based pitch detection, we determined that our formula detected "ballpark" values of the correct frequency (but was not perfect). However, it was easy to distinguish exactly where one note started and the other note ended (if there was a break between notes with the same frequency).

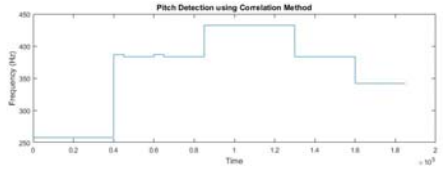


The blips (valleys in the graph) are due to the transients between notes. This was useful to us, because we used the transient values to differentiate between successive notes of the same frequency when creating our matrix of midi values.

The empirical auto-correlation function of x[n] = cos(w0n+ϕ) is given by: [1]

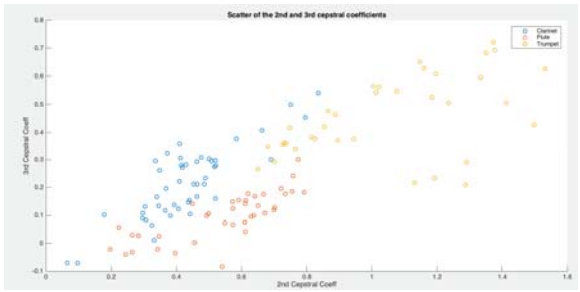
$$\hat{R}[m] = \frac{1}{N}\sum_{n=0}^{N-1-|m|}(w[n]x[n]w[n+|m|]x[n+|m|])$$

Here is a graph of our pitch detection using the auto-correlation based pitch detection method:



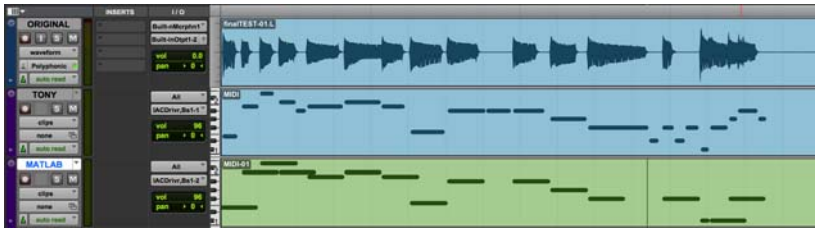
Here we can see near perfect pitch detection. However, we don't have the data to differentiate between successive notes of the same frequency. We used the data from this method and added data from onset detection to create new notes in the MIDI matrix.

Timbre

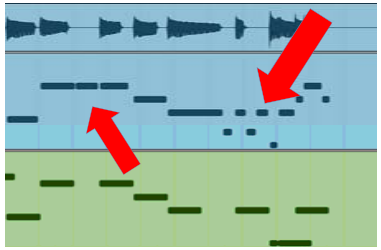


Above is a scatterplot of the second and third cepstral coefficients from three instruments: clarinet, flute and trumpet. As you can see, each instrument occupies it's own space in the quefrency domain. We compiled a library of instruments, analyzed and recorded the general quefrency space that they occupied, and compared them to the cepstral coefficients that came from the input file. We are then able to provide a recommendation for what kind of synthesized instrument should be used with the MIDI file.

Testing and Validation

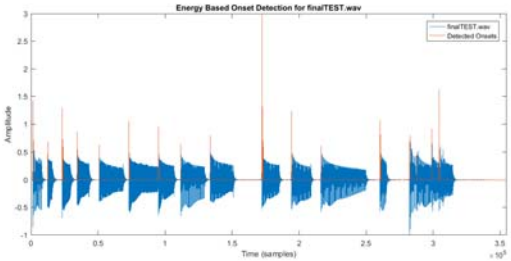


The MATLAB implementation was tested against the output from TONY, a published tool for melody transcription. In the figure above, the waveform of the input audio file is on the top track. The second track contains the outputted MIDI file from TONY. The bottom track contains the output from the MATLAB function. In our preliminary test, the MATLAB function provided a more accurate transcription than TONY did.



In this figure, the TONY MIDI track (middle) contains several errors. For example, MIDI notes were added when there was no sound in the original source file. The MATLAB implementation did not have those errors.

Onset Detection



Since our project is based on monophonic music analysis, a spectral onset detection method proved to be unnecessary. In the figure to the left, you can see the simple energy based onset detection method creating clear and concise onset markers at each onset.

Conclusions and Future Work

The end result was in line with our goal when starting the project. We were able to write and successfully implement a MATLAB program that could perform monophonic audio analysis and MIDI transcription better than the currently available TONY application. Timbre modeling, which is not available in the TONY application, was also successfully implemented.

Future Work: We hope to eventually delve deeper into the concept of polyphonic audio analysis and MIDI transcription. In addition, we would like to eventually write a standalone application separate from MATLAB, which can be provided with audio files, and even create/export synthesized versions.

References

[1] Seo, Naotoshi. "Project: Pitch Detection." *Naotoshi Seo*. N.p., n.d. Web. 02 Apr. 2016.
Schutte, Ken. "MATLAB and MIDI." *Ken Schutte*. N.p., n.d. Web. 02 Apr. 2016.
"Make Music :: Philharmonia Orchestra." Philharmonia Orchestra. N.p., n.d. Web. 20 April 2016.
Anderton, Craig. "Craig Anderton's Brief History Of MIDI." MIDI Association. N.p., n.d. Web. 25 Apr. 2016.

Acknowledgements

We would like to thank Professor Duan for providing us with the resources and support to make this project possible. We'd also like to thank Yuping Iris Ren and Xinzhao Liu for their constant availability and help.