Tune Helper: an auto-tuning software to modify pitch

Zhe Chen, Chenxing Wu, Maxin Chen
EECS 352 - Machine Perception of Music and Audio,
Northwestern University

Introduction and Motivation

Tune Helper is a program, written and built in MATLAB, that can implement functions of "Speech-to-sing", tuning human voice to sing a simple melody, or "Auto-tuning", and creating perfectly tuned vocals by altering off-key pitches.

The project is inspired by music software Auto-Tune, which can auto correct vocals, creating a very cool electronic vocal effect. We want to implement the basic function of Auto-Tune, pitch auto-correction, on MATLAB by ourselves. Also, we want to implement an auto-tuning speech function, which takes a human speech and a melody as inputs, combines human speech with melody, and produces an audio output with a tuned human speech whose pitches are modified to that of the input melody.

Methodology and Algorithm

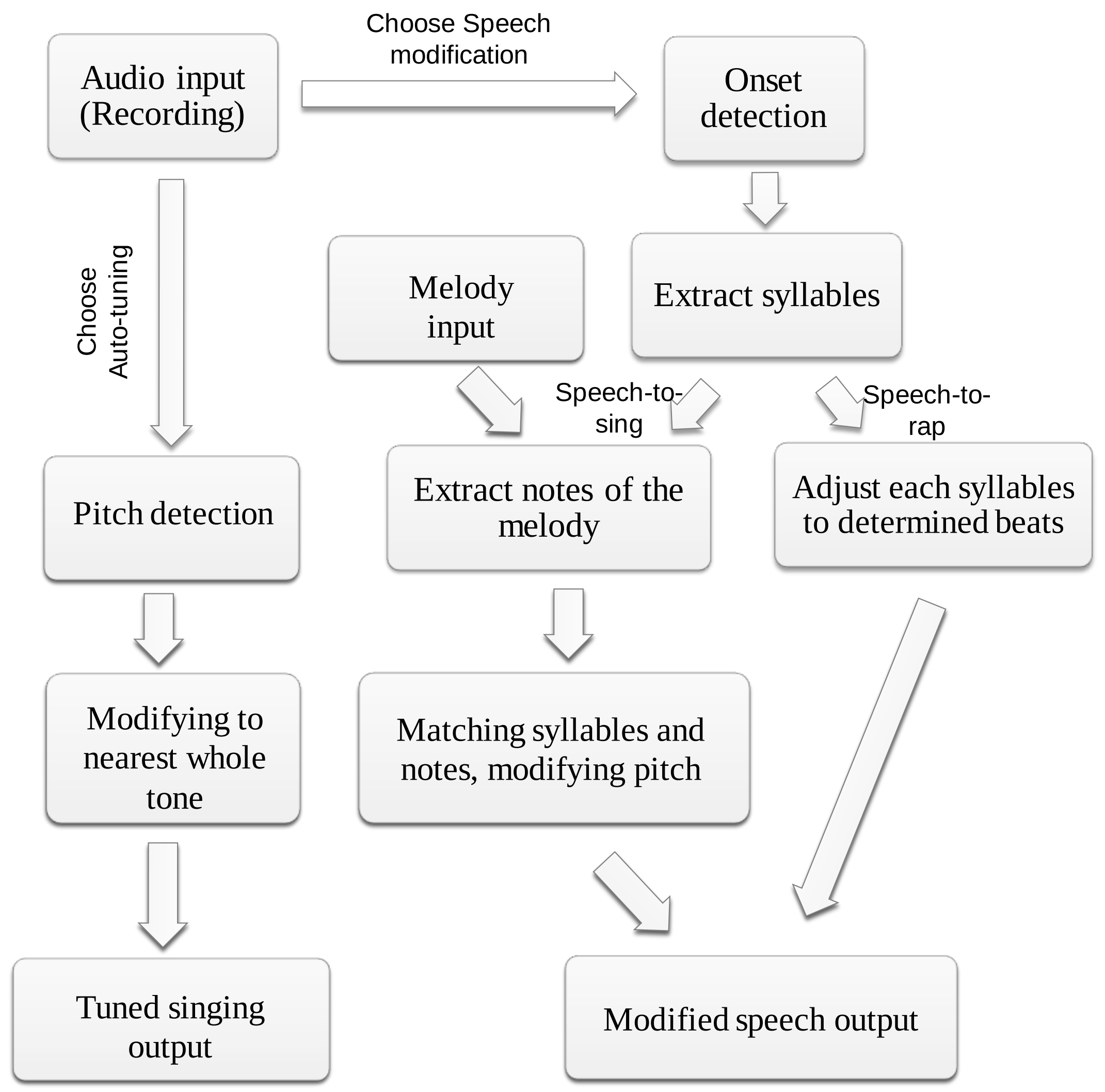
Primary methods used:

Pitch tracking: to extract pitches, time, and amplitude information from input audio;

Beat tracking: to detect onsets and match syllables to notes in a melody;

Phase Vocoder and resampling: to perform time expansion/ compression and pitch shifting of audio clip.

Algorithm:


Audio input (Recording)
Choose Speech modification
Onset detection
Choose Auto-tuning
Melody input
Extract syllables
Speech-to-sing
Speech-to-rap
Pitch detection
Extract notes of the melody
Adjust each syllables to determined beats
Modifying to nearest whole tone
Matching syllables and notes, modifying pitch
Tuned singing output
Modified speech output

Results

Auto-tuning: We input an audio file with some out-of-tune notes on purpose and compare pitches of input and output to see if our program can manage tuning singing. Figure 1 shows comparison of pitches. It is clearly seen that pitches before auto-tuning (blue line) are modified to a perfect pitch.

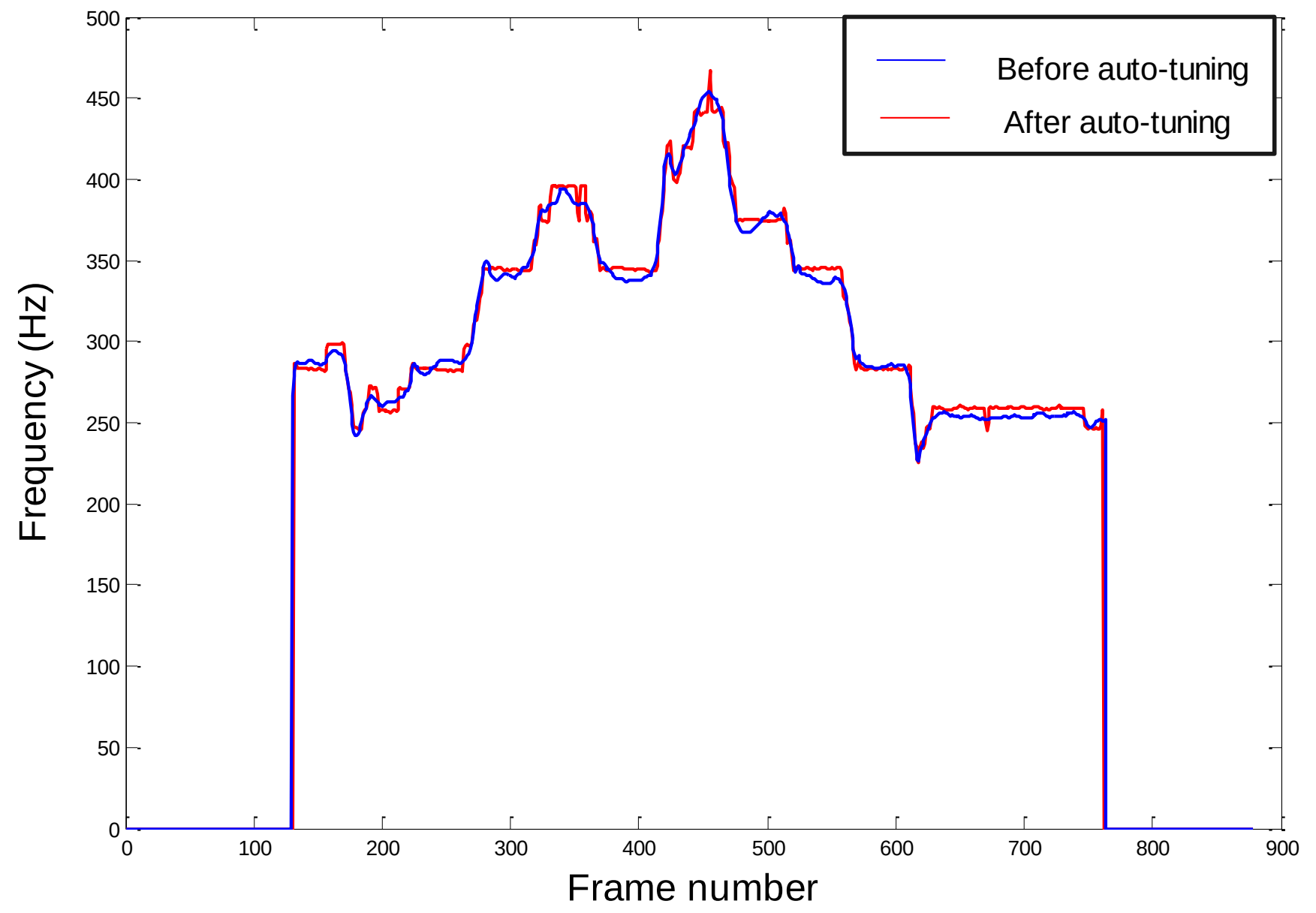
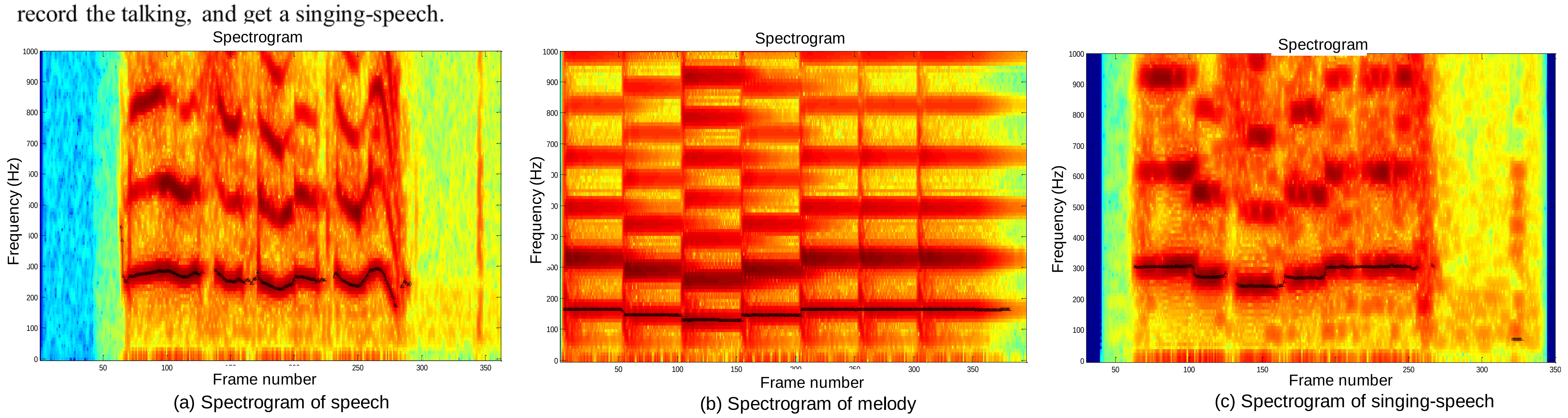


Figure 1 Pitches before and after auto-tuning

Speech-to-sing: We use the famous nursery rhyme "Mary had a little lamb" as an example of the melody input and a recording of reading its lyrics as the speech input. Demo of the inputs and output audio files are available in our website. Figure 2 shows spectrograms of human voice input (a), piano melody input (b), and output audio (c). It shows that each syllable can be detected and shifted to the notes in melody. Using our program, user can import a melody, record the talking, and get a singing-speech.



(a) Spectrogram of speech
(b) Spectrogram of melody
(c) Spectrogram of singing-speech

Figure 2 Spectrogram comparison of audio files in speech-to-sing function

Speech-to-rap: We import a recording of reading a few sentences, process it with "Speech-to-Rap" function, and we have an output of rap accompanied with drum. Input and output demos are available in our website.

User Interface: Figure 3 shows the User Interface of our program. It can load melodies, record voices, and present outputs with an effect followed by users' choices.

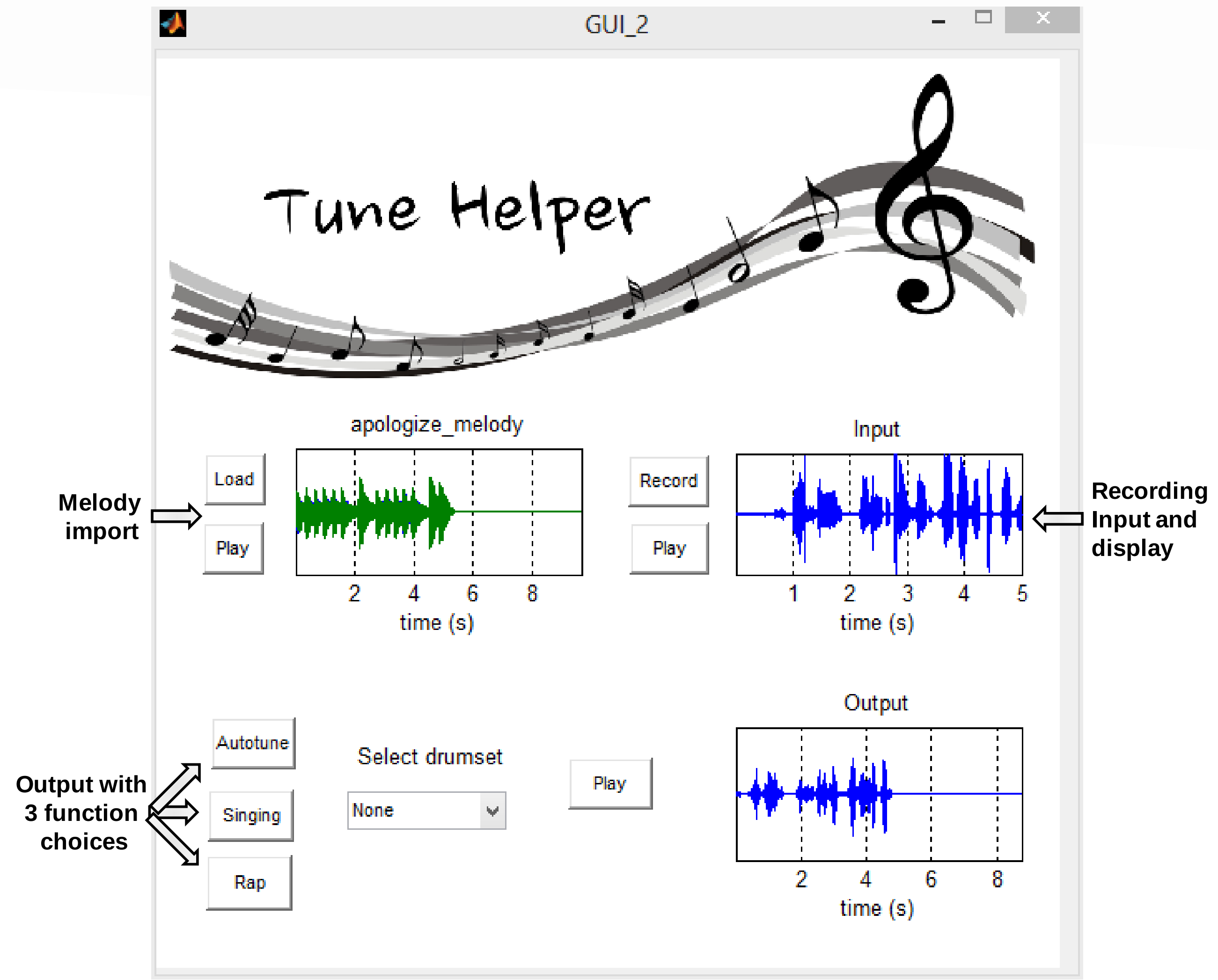


Figure 3 User Interface

Conclusion and Future Work

Our program Tune helper:

- achieved both auto-tuning and speech-to-sing/speech-to-rap functions;
- managed tuning or converting recording to a desired audio output chosen by users.

Future work

- Our program is relatively slow. We need to modify algorithm and codes to shorten processing time.
- We hope to improve speech recognition function to distinguish syllables when the input audio is noisy or the syllables are not really articulated.
- We'd like to add background music in speech-to-sing/rap function to make effects cooler.

References:

[1]. Laroche, Jean, and Mark Dolson. "Improved phase vocoder time-scale modification of audio." Speech and Audio Processing, IEEE Transactions on 7.3 (1999): 323-332.

[2]. Bello, Juan Pablo, Giuliano Monti, and Mark B. Sandler. "Techniques for Automatic Music Transcription." ISMIR. 2000.

[3]. B. McFee and D.P.W. Ellis, "Better beat tracking through robust onset aggregation," in International conference on acoustics, speech and signal processing, 2014, ICASSP.

Acknowledgements

We'd like to thank Professor Bryan Pardo for his suggestions and supports. We'd also like to thank all of the classmates in this course.

Website

http://chenzhe142.github.io/nu-eecs352/