Music Identification

1 Project Approach

Our challenge was to replicate the Shazam music detection algorithm proposed by Avery Wang. Reading Wang's paper, *An Industrial-Strength Audio Search Algorithm*, we realized that an identical approach using hashing and constellation analysis would be tedious and besides the point of the project. Rather, our goal is to learn how Fourier series analysis can be applied to real world problems such as audio analysis. We developed a system where a *.WAV* file can be converted into a constellation (fingerprint) of dominant frequency points across time and then compared to new inputs through correlative analysis. This approach requires time to create the database of initial fingerprints but has very little performance overhead to analyze new files once the database has been completed.

2 Creating Fingerprints

The creation of finger prints to uniquely represent a music file was the most crucial step in the analysis process. Using the *TwinPeaks.m* function from a previous lab, we adapted our code to find not two but any number of dominant peaks in a given sample. Then we generated a windowing system around the peaks locator which scanned the input song and recorded the dominant frequencies in each time period. The fingerprints are automatically saved for analysis and we have included several of them in Figure 1.

We developed these functions to be highly configurable, allowing the technician to determine how many peaks they wanted to save per time period and how many time periods in a song they were willing to save. However these settings must be kept constant lest the comparisons be knocked off track.

3 Comparing Fingerprints

With a database of song fingerprints at our disposal, our next task was to compare them to identify new inputs in relation to our database. There were many methods available to us for this task, including the constellation approach detailed by Wang, but we elected to run a correlative analysis with additional windowing of the fingerprint files, see Figure 2. In more precise terms, we decided to do the following:

- **Create New Fingerprint**
    - Take song name as argument
    - Load *.WAV* file
    - Get fingerprint values
- **Correlative Analysis**
    - Isolate segment of fingerprints
    - Calculate correlation matrix
    - Correlation >= Set tolerance
        - Return the current song
    - Correlation <= Set tolerance
        - Repeat with next song

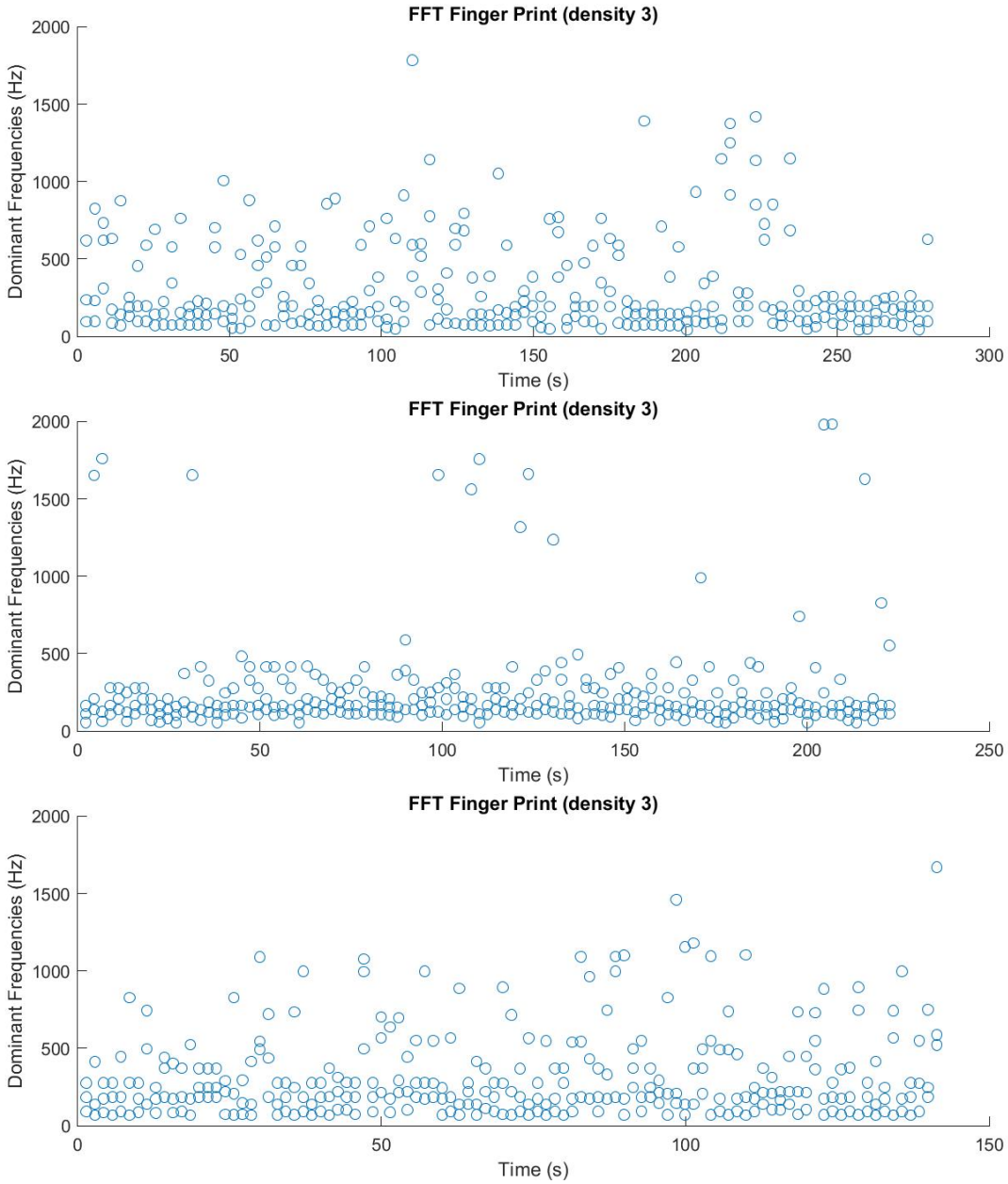

Figure 1: Bob Marley - *I Shot the Sheriff*,
Dire Straights - *Lady Writer*,
Sam Cooke - *Summertime*.

4 Algorithm Results

Our system correctly guesses the correct song every time with a 100% correlation. These results are fantastic and misleading, it only proves that our sampling process does not have stochastic inputs, it is repeatable. In other words, the fact that a 100% correlation occurs reliably is evidence of the fact that the finger print of the input file and the fingerprint in the database file are identical. This is ideal, our system ought to be stable and our results are testimony to that claim, but there is room for improvement to our existing model. For instance, we could scale the sampling so that there are more samples in the beginning of the time range than at the end, allowing for faster identification

of sound files.

We are being too hard on ourselves, the fact remains that this system works perfectly. To test for the robustness against noise, we added a feature to add normalized random values within a specific range to our fingerprints and test if the system fails to detect the signals. Even with aggressive randomness[1], our algorithm had no problem identifying the song correctly.

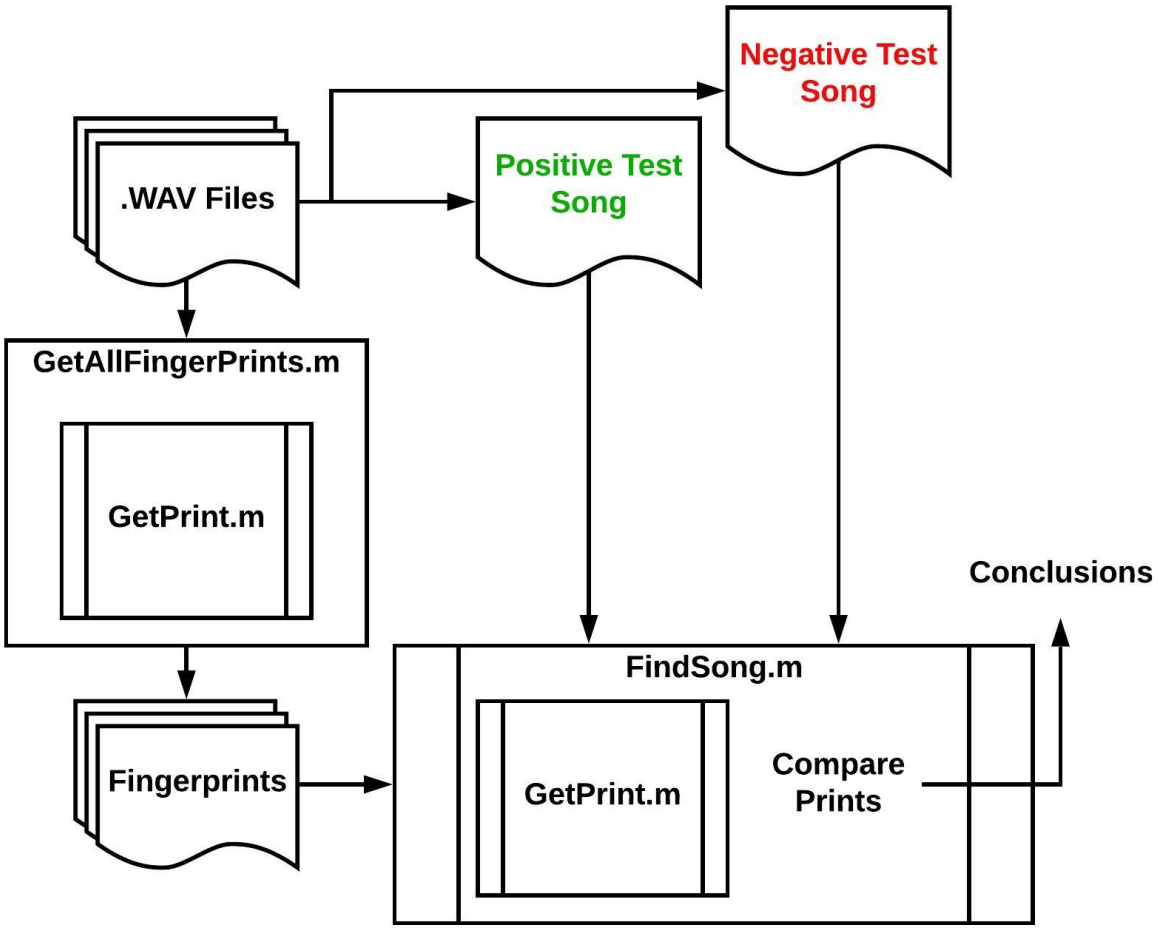

Figure 2: Higher level system architecture.

[1]We used values between 0 and 200 and achieved $>$ 90% accuracy.