AUDIO MOSAICING WITH FREESOUND

Logan Stillings
Universitat Pompeu Fabra
loganstillings01@estudiant.upf.edu

ABSTRACT

The goal of audio mosaicing is to generate a mosaic recording that conveys musical aspects (like melody and rhythm) of the target recording, using sound components taken from a source recording [3]. In this work, frames of the target records are matched with similar frames from the source audio by using a K-Nearest Neighbor (KNN) algorithm based on spectral and harmonic features. I modified the provided jupyter notebooks to perform five different audio mosaics using various target recordings with the same set of source queries.

1. INTRODUCTION

1.1 Target Audio Files

Five files were chosen from Freesound [4] from distinct musical genres. Multiple genres were chosen to observe potential differences in the way automated mosaicing occurs in different types of music. The target audio can best be described as an 80's rock beat, a music-box lullaby, Turkish classical music, a drum loop, and a 2000's pop beat. Each target track is between 20 and 40 seconds in length.

1.2 Source Audio Files

The source audio files come from four different Freesound search queries. Each query contains a search term, a filter, a maximum number of results, and a property for sorting the top results. The first two queries use search terms of different instruments, namely the guitar, and violin. Each of these instrumental queries implements a filter such that the key of the source matches the estimated key of the target audio, and that the tonality confidence for the source is greater than 90%. The instrumental queries return the top 20 results and are sorted by the most downloaded sounds on Freesound. In addition to the instrumental search queries, I included two other types of atonal sounds. The first atonal sound uses the search term "wilhelm", which is used to retrieve a sound of the Wilhelm Scream, a stock sound effect used commonly in film and television. The second atonal search uses the search term "dog bark". Each of these atonal searches get the first 10 results, and are sorted based on score and most downloaded sounds respectively. The search queries remain the same for each of the five target files, however the differences in estimated key for each of the targets will result in a slightly different pool of source sounds.

2. METHODOLOGY

In order to perform audio mosaicing on the target audio from the source collection, I implemented the provided jupyter notebooks with several enhancements. The first enhancement was to extract tonality from the target audio and find source units with the same tonality. To do so, I implemented the KeyExtractor algorithm from the Essentia [1] library. The KeyExtractor algorithm computes a Harmonic Pitch Class Profile (HPCP) [5] from the spectral peaks at frames of the input signal and applies key estimation. Once I obtain the estimated key of the target signal, I apply that key as a filter for the source audio search query for the instrumental queries.

The next enhancement I made to the original was to extract beat positions from the target audio files so that the reconstructed mosaic would contain fragments that match the beats of the original audio. To do so I implemented Essentia's BeatTrackerDegara [2] algorithm, which computes a 'complex spectral difference' onset detection function and utilizes the beat tracking algorithm (TempoTapDegara) to extract beats. I replaced the static frame size in the frame and source analysis section with frames that are the size of the beat.

Also in the code for analyzing the source audio, I included key, key strength and scale as features. These features are calculated using the same KeyExtractor algorithm mentioned in the section for creating the source collection from Freesound. One important step done here is encoding the keys and scales as integer values that could be understood by the K-Nearest Neighbors algorithm implemented in the similarity calculation for mosaic reconstruction. I decided to save the encoded values in the data frame itself instead of processing the categorical data before running the knn algorithm for the sake of convenience.

Once finding the ten frames of the source collection that are most similar to a frame of the target audio, I implement a random distribution to select the frame to use for the mosaic. The most similar frame has a 55% chance of being selected and the second through tenth most similar frames each have a 5% chance of being selected. I used python's random module to achieve this. The randomness was added to have move variety in the source frame that is

selected especially if the target audio contains some repetition across beats.

The last enhancement I made to the original notebooks was to save the mixed signal as its own file, as I prefer the resulting sound of the mixed version to the reconstructed version alone.

3. RESULTS

In this section I will go over the generated mosaics for each of the target sounds implemented.

3.1 Turkish Classical Music

The resulting mosaic for the Turkish Classical Music target contains two guitar sounds, one violin sound, one wilhelm scream sound, and four dog barking sounds. It is a bit strange that the dog barking sounds were often considered within the ten most similar frames of the original target, given the harmonic nature of the music. One possible explanation of this could be the inclusion of key strength as a feature for similarity. Turkish classical music contains micro tonalities that could result in low key strength as calculated by the KeyExtractor algorithm. Thus that would explain why an atonal sound like dogs barking would be considered similar. Otherwise the inclusion of guitar and violin sounds in the reconstruction makes sense due to the presence of a stringed instrument in the target audio.

3.2 Music-box Lullaby

The mosaic for the Music-box lullaby target contains one wilhelm scream sound and five dog barking sounds. It was a bit peculiar that the lullaby would not contain any of the instrumental tracks, but upon further investigation, I realized that the source collection only contained the screams and the dogs, without any of the instruments. The estimated key of the Music-box Lullaby was Ab Major, however the Freesound query using this key was unable to find any guitar or violin sounds that matched this key with high confidence. This is a possible downside of trying to automate the audio mosaicing task for targets of any key, simply because the source dataset may not contain enough examples of the desired key.

3.3 Drum Loop

The mosaic for the drum loop target contains fragments from two guitar sounds, three scream sounds, and three dog barking sounds. The estimated key of the original target was F major (with a confidence of 81%), of which Freesound was able to find 13 guitar sounds, however there were zero violin sounds in the source collection. Interestingly, the mosaic is mostly made up of a repetitive guitar playing F over the original drum loop. My initial reaction was that these results conflicted with my theory of why there were so many dog barking sounds in the Turkish classical music mosaic, however I was surprised to see that the KeyExtractor algorithm had high confidence in estimating the key for the drum loop despite little tonal information.

3.4 80's Rock Beat

The mosaic for the 80's rock beat contains fragments from three guitar sounds, one violin sound, three scream sounds, and one dog barking sound. The balance of instruments used in from the source collection for this track was a lot closer to what I wanted for each of the previous mosaics. The estimated key of the target song is C minor with 87% confidence. The result does become somewhat chaotic in the middle of the mosaic, though I believe this is something that could be fixed with more robust beat tracking applications in which the source audio frames are also of similar tempo to the target.

3.5 2000's Pop Beat

The mosaic for the 80's rock beat contains fragments from seven guitar sounds, one violin sound, one scream sound, and two dog barking sounds. The balance of instruments in the source collection for this mosaic was desirable. The guitar sounds make up most of the reconstructed sound, but the beat matching seems better than that previously mentioned in the 80's rock beat mosaic.

4. DISCUSSION

To reiterate some of the points mentioned above, I believe that this type of work could benefit with further experimentation on microtonal targets, with lower tonality confidences for instrumental sources to increase the number of sounds in the source collection, and to use more robust methods of beat matching between source and target audio. Please note that when reproducing these results, the output files will be slightly different each time due to the randomness implemented in the selection of the most similar frames.

5. REFERENCES

[1] Bogdanov D, Wack N, Gómez E, Gulati S, Herrera P, and Mayor O. Essentia: an audio analysis library for music information retrieval. *International Society for Music Information Retrieval Conference*, pages 493–498, 2013.

[2] N. Degara, E. A. Rua, A. Pena, S. Torres-Guijarro, M. E. Davies, and M. D. Plumbley. Reliability-informed beat tracking of musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*, pages 290–301, 2012.

[3] Jonathan Driedger, Thomas Pratzlich, and Meinard Muller. Let it bee – towards nmf-inspired audio mosaicing. *16th International Society for Music Information Retrieval Conference*, 2015.

[4] Frederic Font, Gerard Roma, and Xavier Serra. Freesound technical demo. *Proceedings of the 21st ACM international conference on Multimedia*, 2013.

[5] E Gómez. Tonal description of polyphonic audio for music content processing. *INFORMS Journal on Computing*, pages 294–304, 2006.