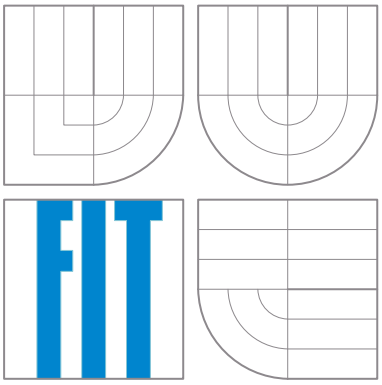

Multimedia

Automatic audio splitter

May 3, 2017

Authors: Ján Profant, xprofa00@stud.fit.vutbr.cz
Daniel Bambušek, xbambu04@stud.fit.vutbr.cz

Faculty of Information Technology
Brno University of Technology

Contents

Assignment

Create and implement an application, which is able to split very long *.wav* file to independent tracks. Motivation is in splitting digitalized vinyl album to tracks, parametrized by length of silence between tracks and normalized level of silence.

Approach

We chose to implement our solution in python2 and also create GUI in PyQt4 [5]. We used approach similar to Voice Activity Detection (denoted as VAD for the rest of this paper).

VAD

A voice activity detector is a device (program) which is supplied with a signal with the object of detecting periods of speech, or periods containing only noise. Although the present invention is not limited thereto, one application of particular interest for such detectors is in mobile radio telephone systems where the knowledge as to the presence or otherwise of speech can be used and exploited by a speech coder to improve the efficient utilization of radio spectrum, and where also the noise level is likely to be high [4].

We read input *.wav* file using scipy.io.wavfile [1]. We extract frames and use following approach:

- Compute normalized energy
- Compute band energy defined by band start and end
- Compute ratio between them
- Compare it with threshold and alternatively, use median filter for smoothing

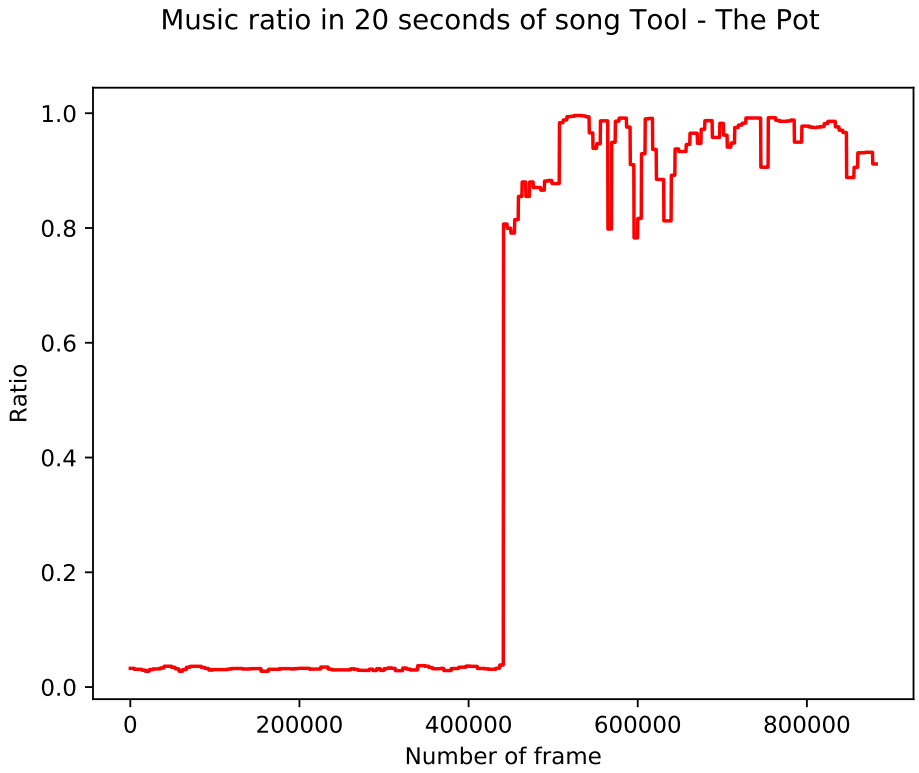

Figure 1: Visualization of music ratio in song Tool - The Pot with 10 seconds of silence at the beginning.

For parallel processing we used python library called joblib, basically we chunk the input data and run it independently. At the end, we merge them together and run post-processing for detecting tracks.

Application Interface

We developed GUI application with simple interface 3, which is able to:

- Browse input file
- Set the input parameters, such as silence level, silence length, band start, band end, minimal length of song, usage of median filter and number of cores to use
- Detect tracks in input audio file
- Modify detected tracks (name, start and end of the song), play them
- Save them as separated tracks

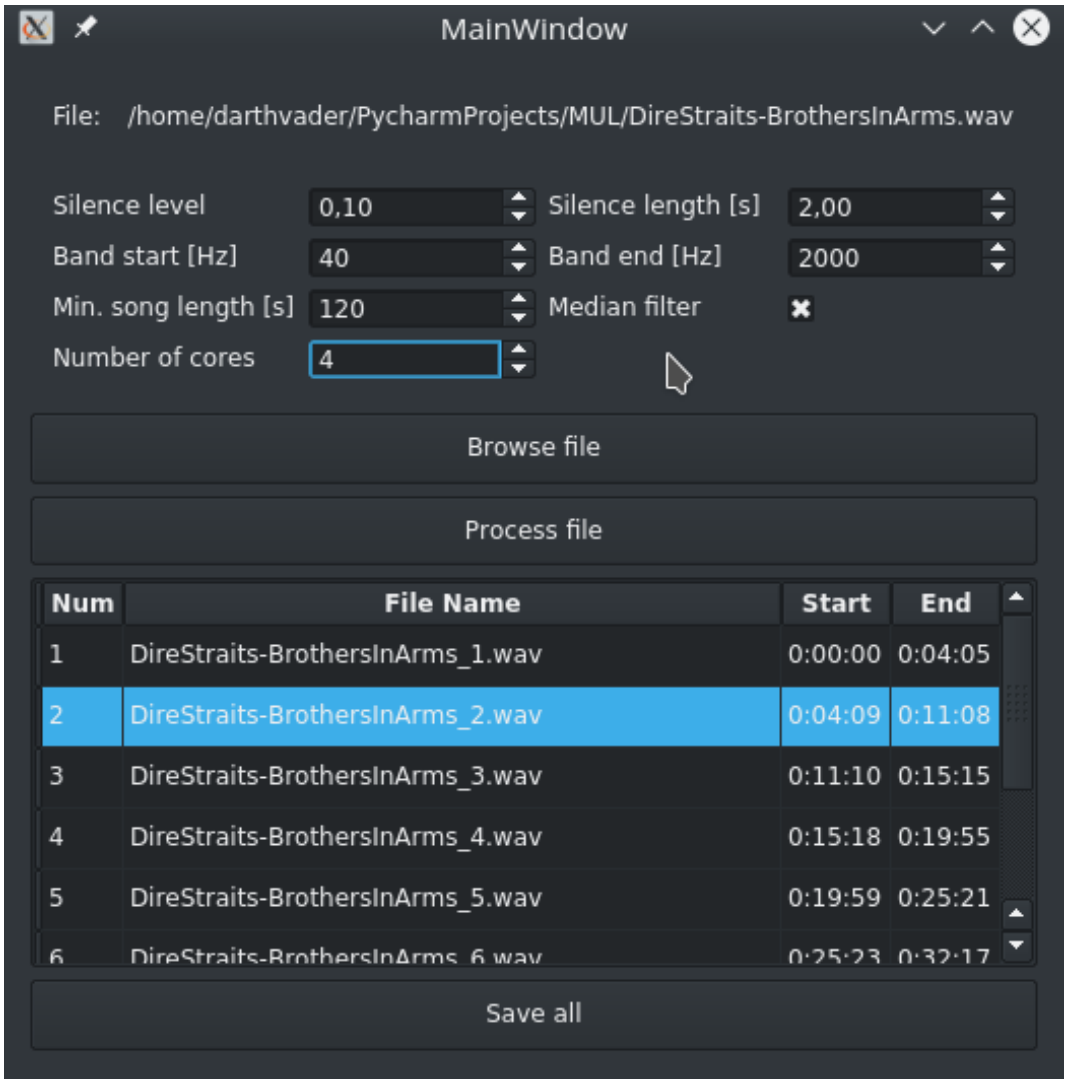

Figure 2: GUI of application after processing vinyl digitalized album - Dire Straits - Brothers in Arms.

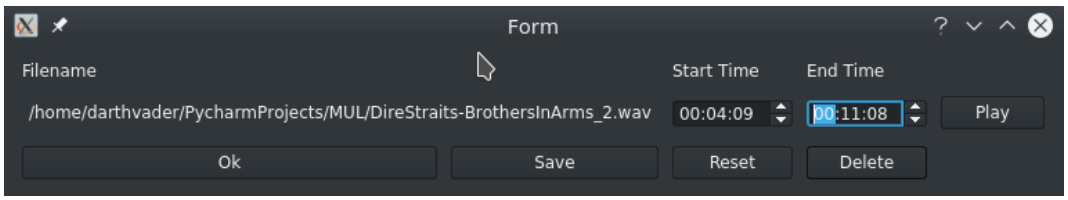

Figure 3: Editing form is able to change name, save, reset, play and modify start and end of track.

Dependencies

These are the python2.7 dependencies:

- numpy
- scipy

- multiprocessing
- joblib
- sounddevice

Demonstration

In our demonstration, we used album by Dire Straits - Brothers in Arms [3]. Vinyl version of album consists from 9 tracks and has been downloaded using youtube-dl [2]. We found out, that space of parameters is quite large, so we ran more than five thousand experiments to find best setup. Of course, parameters can be too specific for this example, but we can see, that it works quite fine even for other examples. Reference times were defined by listening to the album (times may vary from origin). We achieved 40 seconds difference in total using our application, it was able to find every song, we can see detailed result in figure 4. It can look like poor result, but we must consider, that even for human it is quite difficult to specify, when the song ended and when the other started - especially because slow muting specific for vinyl album. Our application is also quite slow, we achieved 17.5 FTR (faster than realtime) on single core, when using 4 cores 45 FTR - i7-4710HQ.

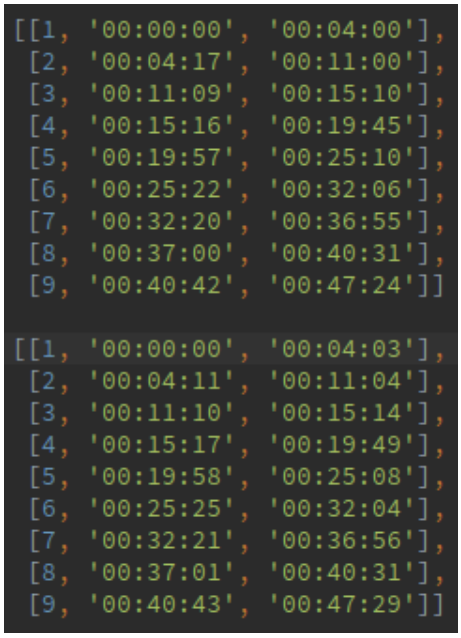

Figure 4: Comparison of reference segmentation (up) and output from our application (down).

Sources

Source codes are well documentedSource code in project root directory:

- *main.py* - main GUI application, definition of GUI and PyQt classes
- *vad.py* - VAD in python

- *nogui.py* - application without using GUI, process input file and print result
- *profile.py* - script used for searching best parameters
- *mainwindow.ui* - UI of main window created in QtCreator
- *main_window.py* - UI class in python automatically generated from .ui file
- *form.ui* - UI edit form created in QtCreator
- *form.py* - UI automatically generated from .ui file

Conclusions

We were able to implement simple application with GUI, which can automatically detect tracks in digitalized vinyl album. We estimated parameters according to our method - VAD, which is more suitable for detecting speech segments, but is able to work also in this case. When comparing results to reference, we achieved only 40 seconds difference in 47 minutes long album in track segmentation.

Bibliography

[1] SciPy Developers. SciPy.
https://www.scipy.org/.

[2] Collection of authors. Download videos from YouTube (and more sites).
https://rg3.github.io/youtube-dl/.

[3] Dire Straits. Dire Straits - Brothers in Arms.
https://www.discogs.com/Dire-Straits-Brothers-In-Arms/master/23684.

[4] Daniel K Freeman and Ivan Boyd. Voice activity detection, January 4 1994. US Patent 5,276,765.

[5] Phil Thompson. PyQt4.
https://pypi.python.org/pypi/PyQt4.