MIR Assignment 2, Spring 2016 (10 pts)

The goal of this assignment is to familiarize you with monophonic pitch detection and simple audio feature extraction - namely the extraction of a centroid. This assignment requires some programming. You can choose any programming language/environment you like but some of the questions are specifically designed for *Marsyas* and will be easier if you use the framework. You can choose to use the *Marsyas* scripting language, C++ code, or the Python bindings. Do not hesitate to contact me with any questions you might have.

The assignment is worth 10% of the final grade. There is some variance in the amount of time each question probably will require. Therefore dont expect them to be equally difficult even though they are all worth the same number of points. Please provide your answers in a SINGLE PDF file. There is no need to copy the assignment specification. Also please answer the questions in order and explicitly mention if you have decided to skip a question. For questions that involve programming provide a listing of the relevant code but not all the details that are not directly relevant to the question.

Hope you find it interesting, George Tzanetakis

1 Monophonic pitch estimation (5 points)

The goal in this question is to extract a fundamental frequency contour from an audio file. Utilize the *qbh-examples.wav* audio file available on the course webpage for testing. You are welcome to implement this in any environment/language you like but it has been designed to provide a gentle introduction to *Marsyas* an open-source audio analysis framework specifically designed for Music Information Retrieval. In *Marsyas* algorithms are expressed as networks of processing objects with data passing through them (similarly to block diagrams in signal processing). It is possible to work directly in C++, or using Python bindings in Python, or use the custom-designed *Marsyas* scripting language.

- **(2 point) (*)** A simple fundamental frequency estimation is to compute the magnitude spectrum, select the highest peak, and return the corresponding frequency as the result. Processing the sound in windows will result in a time series of F0 estimates that can be plotted over time. In *Marsyas* the following *MarSystems* would need to be connected: *SoundFileSource*, *Windowing*, *Spectrum*, *PowerSpectrum*, *Selector*, *MaxArgMax* and *Accumulator*. Check that your method works by using as input a sine. Show the F0 plot for *qbh_examples.wav* with this method when using windows of 1024 samples at 22050 sampling rate.

- **(2 point) (*)** Copy your network and modify it to use *Autocorrelation* for the f0-estimation. Plot the new contour and compare it with the previous one. Is one consistently better than the other ?

- **(1 point) (**)** Change your code so that both the F0 estimates are computed for every window. Plot the sum of the two resulting F0 contours. The computation should be done in one pass through the audio file. In *Marsyas* use a *Fanout* after the *Windowing* followed by a *Sum*.

2 Centroid Sonification (5 points)

The purpose of this question is to experiment with the audio feature called *spectral centroid* using *Marsyas*.

2.1 (1 point) (*) Preparation

Download and install *Marsyas*, and learn the basics with the help of the online tutorial: `http://marsyas.info/tutorial/tutorial.html`. Don't hesitate to contact your course instructor or TA for help.

In the following steps you will create a *Marsyas* script that reads an audio file, computes the spectral centroid of each window of the short-term Fourier transform (STFT), and plays back both the original audio file and a sine wave that follows the spectral centroid frequency. The structure of the script will be very much like the example in the "Control links" section of the tutorial: `http://marsyas.info/tutorial/tutorial.html#control-links`. However, instead of applying the RMS energy to the amplitude of a noise generator, you will apply the spectral centroid to the frequency of a sine wave generator. It is a good idea to start with the example code and modify it to achieve the goal of this question.

Note that the same experimentation can be done with the *Marsyas* Python bindings or the *Marsyas* C++ library, and you are welcome to do so if you prefer.

In any case, please provide your entire code for this question.

2.2 (2 points) (**) Compute the spectral centroid

This step replaces the RMS energy computation. First compute the STFT with a window size of 512 samples and 1/2 window overlap. Do this by combining a ShiftInput, a Spectrum, and a PowerSpectrum MarSystem. Then add a Centroid MarSystem. Smooth the centroid stream by applying an "audio feature texture window": compute the mean of the last 20 values using a Memory and a Mean MarSystem. Finally, convert the result to a control stream using a FlowToControl MarSystem.

2.3 (1 point) (**) Apply the centroid to the frequency of a sine wave

This step replaces the playback of noise that follows the RMS energy. Use a SineSource MarSystem to generate the sine wave. Create a link between the centroid control stream and the frequency control of the SineSource. However, the Centroid MarSystem produces a number between 0 and the number of power spectrum bins. This needs to be converted to a frequency value in Hz between 0 and the Nyquist frequency. Use a control expression to do the conversion using the current sample rate. The sample rate is available as the "israte" control of any MarSystem (in this case SineSource itself).

2.4 (1 points) (***) Experiment with music genres

Download one or two classical music files and metal music files from the corresponding folder of: http://marsyas.cs.uvic.ca/sound/genres/

Run two instances of your centroid sonification script at the same time (using two command-line windows) to compare the results of pairs of different input audio files:

- classical and classical
- classical and metal
- metal and metal

To some extent this would be what an automatic genre classification system based purely on the Spectral Centroid would "hear" to make a decision of classical or metal. FYI using just the mean of the spectral centroid a trained classifier makes the right decision 75% of the time. This is impressive given that decisions are made every 20 milliseconds so even better results could be obtained by some majority voting/filtering over the entire file.

To see the influence of the texture window comment out the MarSystems Memory and Mean. Provide a very brief commentary (no more than 3-4 sentences) of your observations on this sonification experiment.