MIR Assignment 3, Spring 2015 (100 pts)

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 due on March 18 (soft deadline).

IMPORTANT Some questions have additional deliverables for the graduate students taking the course. These are marked with [CSC575]. Undergraduate students are welcome to do them but they are optional and will not affect their grade. 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. You can either hand me a paper copy of your submission in class or email me an electronic copy.

Hope you find it interesting, George Tzanetakis

1 Centroid Sonification (35 points)

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

1.1 (10 points) 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.

1.2 (10 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.

1.3 (10 points) 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).

1.4 (5 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.

1.5 [CSC575]

Extend your code so that it plays two audio files and their sonified centroids at once.

2 Classification using Audio Features (30 points)

Download the 1.2 GB genre classification dataset from:
`http://marsyas.info/download/data_sets`
You will only need 1.2 GB of space for download but after that you can pick any three genres out of the 10 genres for your experiments. Alternatively if you don't have enough space you can download individual files for 3 genres (at least 20 tracks for each genre) from:
`http://marsyas.cs.uvic.ca/sound/genres/`
Read the instructions in Chapter 3 of the Marsyas User Manual (Tour - Command Line Tools) and use the **bextract** command-line program to extract features for the 3 genres you selected. Load the extracted .arff file into Weka and report on the classification accuracy of the following classifiers: ZeroR, NaiveBayesSimple, J48, and SMO.

Your deliverable will be the list of command you used and the classification accuracy + confusion matrix for each classifier for the 3-genre experiment.
[CSC575] The command-line options to bextract -ws, -hs control the analysis window size and hop size for the audio feature calculation. Create a table showing how classification accuracy is affected by that choice. Investigate all combinations of the following values 256, 512, 1024, 2048, 4096 for each. For example 256 hop size and 1024 window size, or 1024 hop size and 1024 window size.

3 Reading (35 points)

Read the paper "Semantic Annotation and Retrieval of Music and Sound Effects"
`http://cosmal.ucsd.edu/~lbarring/pubs/`
`Turnbull,Barrington,Torres&Lanckriet-SemanticAnnotation&RetrievalOfMusic.pdf`
which is one of the first published works exploring automatic music tagging and answer the following questions:

1. **(15 points)** In Section IIIB the authors formulate the annotation problem using probabilistic notation. Come up with a concrete example with made up numbers where each song has a sequence of 4 feature vectors of dimension 2.

2. **(10)** The authors describe three parameter estimation techniques: direct estimation, model averaging, and mixture hierachies estimation. For which category of tags does model averaging work better than mixture hiearchies ? Which category was modeled most successfully using the proposed approach ?

3. **(10)** Describe the differences between the music data set and the sound effect dataset. For which one does auto-tagging seem to work better ? Speculate why this is the case.

4. [**CSC475**] Read about the EM-algorithm and how it can be used for estimating the parameters of a Gaussian mixture model. Prepare a short (2-3 paragraphs) description targeted to undergraduate students for helping them understand how it works. Try to go beyond a dry mathematical description using one or more of the following techniques: 1) code example, 2) animation, 3) static visualization, 4) concrete specific example.