Multimedia
Module No: CM3106
Laboratory Worksheet Lab 2 (Week 3): MATLAB Nyquist's Sampling Theorem, Time-Frequency Analysis

Prof. D. Marshall

After working through this worksheet you should be familiar with:

- The basic function of Nyquist's Sampling Theorem
- Time-Frequency analysis of audio.
- Applications of Short Time Fourier Transform processing.
- Applications of the Phase Vocoder.

None of the work here is part of the assessed coursework for this module ALTHOUGH many of the exercises below will help in parts of of your solution for the assessed coursework

MATLAB Basic Digital Signal Processing

1. **Nyquist's Sampling Theorem**: Download the MATLAB Nyquist sampling frequency demo code from *http://www.cs.cf.ac.uk/Dave/Multimedia/Lecture_Examples/Basic_DSP/nyquist_aliasdemo.m* and the *nyquist_aliasaudiodemo.m*.

   Experiment by changing the sample frequency `samp_freq` and the frequency of the sine wave `freq` and note the displayed waveform and audio results when running the program(s).

   (a) Set the waveform frequency to be *below* the Nyquist limit and notice the effect.

   (b) Set the waveform frequency to be *at* the Nyquist limit and notice the effect.

   (c) Set the waveform frequency to be *just above* the Nyquist limit and notice the effect.

   (d) Set the waveform frequency to be *halfway* between the Nyquist limit and the sample frequency and notice the effect.

   (e) Set the waveform frequency to be *at* the sample frequency and notice the effect.

MATLAB Time-Frequency Analysis

1. **Short Time Fourier Transform**: Load up some audio in MATLAB.

   (a) By varying the size of the the number of sample points for the FFT, the window and the offset of the stft.m code. Compute, display and observe the resultant Spectrograms. See example: stft_spectrogram.m

   (b) By setting $w = 0$ in stft.m observe the difference between a rectangular and hann-windowed STFTs. Display the resultant Spectrograms and also inverse STFT to recover waveforms and display/play these.

   (c) Perform some simple block editing of the STFT:

      i. Cut out a series of **horizontal rectangular blocks** and observe (**plot** and **play sound**) the resultant **Spectrograms** and the inverse STFT's waveforms. Plot and play the **residual** waveform from the difference of the original and modified waveform.

      ii. Cut out a series of **horizontal rectangular blocks** and perform the **above analysis**.

      iii. Cut out a series of horizontal and rectangular blocks and perform the **above analysis**.

      iv. Cut out a series random regions and perform the **above analysis**.

      v. Enhance or diminish the **energy** on horizontal/vertical block and perform the **above analysis**.

      vi. Change the phase by $45°$, $90°$ and some arbitrary huber (remember Phasors from last week's tutorial) of the output STFT, inverse STFT and compare the original and new waveforms.

2. **The Phase Vocoder**: Load up some audio in MATLAB. See example code: pvoc_speed.m:

   (a) **Tempo Change**: Change the speed of the audio by
      i. One half speed slower.
      ii. One half seed faster.
      iii. Three time as fast.

   (b) **Pitch Shifting** With reference to *Meantone intervals* and the example code, pvoc_pitch.m, change the pitch of the audio by:
      i. One semi-tone up from the original pitch.
      ii. One semi-tone down from the original pitch.
      iii. One major third up from the original pitch.
      iv. One major third down from the original pitch.
      v. Create a *thee-part harmony* of the original audio (part 1), the audio shifted up a fifth (part 2) and the audio shifted up an octave (part 3).

   (c) **Advance Phase Vocoding**:
      i. Modify the calling of the `pvsample()` to allow for **freezing** for a portion(s) of audio.
      ii. Modify the calling of the `pvsample()` to allow for **reversal** for a portion(s) of audio.
      iii. Modify the calling of the `pvsample()` to allow for **repetition** (Audio Stutter) for a portion(s) of audio.
      iv. Modify the calling of the `pvsample()` to allow for **modulation** of the phase for a portion(s) of audio.