Task 2 – Sampling-Theory Studio

Introduction: Sampling an analog signal is a crucial step for any digital signal processing system. The Nyquist–Shannon sampling theorem guarantees a full recovery of the signal when sampling with a frequency larger than or equal to the bandwidth of the signal (or double the maximum frequency in case of real signals).

Description: Develop a desktop application that illustrates the signal sampling and recovery showing the importance and validation of the Nyquist rate. You application should have the following features:

- **Sample & Recover:** Allow the user to load a mid-length signal (around 1000 points length), visualize and sample it via different frequencies, then use the sampled points to recover the original signal using Whittaker–Shannon interpolation formula (please, refer to this wiki page for more information). Sampling frequency should be shown in either actual frequency or normalized one (with respect to the maximum frequency, i.e. ranges from $0 \times f_{max}$ to $4 \times f_{max}$ for example). You should use three graphs, one for displaying the original signal along with the markers for the sampled points and another one to display the reconstructed signal, and lastly the third one should display the difference between the original signal and the reconstructed one.
- **Load & Compose:** The loaded signal can come a file or a signal mixer/composer in your application. In the signal mixer, the user can add multiple sinusoidal signals of different frequencies and magnitudes. The user can also remove any of the components during preparing the mixed signal.
- **Additive noise:** The user can add noise to the loaded signal with custom/controllable SNR level. The program should be able to show the dependency of the noise effect on the signal frequency.
- **Real-time:** Sampling and recovery should be done in real time upon user changes (i.e. no button called “update” or “Refresh” is allowed).
- **Resize:** The application can be resized easily without messing up your UI.
- **Different sampling scenarios:** In your submission, you need to prepare at least 3 testing synthetic signals (generated and saved through your Composer) that address different testing scenarios. One example test case is the one mentioned in the lecture:
    - A mix (i.e. summation) of 2Hz and 6Hz sinusoidals. If sampled with 12Hz or above, then it should be recovered nicely. But if sampled with 4Hz, then the two frequencies will show up as just one frequency. What if the signal is sampled with 8Hz?

  You need to provide 3 more examples; each would exploit a problem or illustrate a tricky feature.

Code practice:

- Same practices from Task 1 (i.e. proper variable and functions names) will continue with task 2.