Mini-Project 2

Convolution Reverb Audio Processor

2.1 Synopsis

Discrete convolution serves as the foundation of a popular audio processing technique called *convolution reverberation* or "convolution reverb" for short. Convolution reverb uses the impulse response (IR) recorded within a physical room or hall to artificially add reverberation to an audio signal such as an instrument or voice.

In this lab project you will gain familiarity with the LabVIEW built-in subVI Convolution as a means to convolve small sequences, and then apply this subVI to implement your own convolution reverb audio processor.

Ulaby/Yagle Section 7-5

2.2 Objectives

1. Evaluate the discrete-time convolution sum by hand
2. Confirm by-hand calculations with the LabVIEW Convolution subVI
3. Implement a convolution reverb processor using speech and recorded impulse response (IR) audio files

2.3 Deliverables

1. Hardcopy of all LabVIEW block diagrams and front panels that you create

2. .zip file containing all audio files that you create
3. Lab report or notebook formatted according to instructor's requirements

2.4 Required Resources

1. NI LabVIEW with MathScript RT Module
2. Speech .wav file from the Open Speech Repository, `http://www.voiptroubleshooter.com/open_speech`
3. Impulse response .wav file from Open Air, `http://www.openairlib.net`
4. Resample Audio .wav VI `http://decibel.ni.com/content/docs/DOC-23105`

2.5 Preparation

By-hand computation of discrete-time convolution

As an initial warm-up on the subject of discrete-time convolution, view this short video tutorial `http://youtu.be/yyTu0SXeW1M` to learn how to convolve two sequences by hand. Apply what you have learned by convolving the following pairs of sequences:

$\square_1 \quad \{\underline{1}, 2, 3\} * \{\underline{3}, 2, 1\}$

$\square_2 \quad \{\underline{0}, 0, 1, 1, 1, 1, 1\} * \{\underline{0.2}, 0.2, 0.2, 0.2, 0.2\}$

$\square_3 \quad \{\underline{0}, 0, 1, 1, 1, 1, 1\} * \{\underline{-1}, 1\}$

$\square_4 \quad \{\underline{1}, 2, 3\} * \{\underline{1}, 0, 0, 0, 0, 0, 0, 0.75, 0, 0, 0, 0, 0, 0.33, 0, 0, 0, 0, 0, 0.1\}$

Confirm by-hand computation

► Follow along with this video tutorial `http://youtu.be/Fsf88Rmimz4` to learn how to create a LabVIEW VI to evaluate the convolution of two sequences, and to visualize the input and output sequences as arrays and as stem plots.

$\square_5$ Run your VI for each of the sequence pairs above. TIP: Delete an element from the array by right-clicking on the element and choosing "Data Operations | Delete Element." Collect a screen shot of the front panel for each sequence pair, and then compare to your by-hand calculations. Reconcile any differences between the two by correcting your by-hand calculation or by revising your LabVIEW code.

Download and prepare audio files

► Visit the Open Speech Repository, `http://www.voiptroubleshooter.com/open_speech`. Follow the link for your language of choice, and then download one or more of the speech audio clips.

► Visit Open Air, `http://www.openairlib.net`. Follow the "IR Data" tab at the top of the page and investigate the available rooms and halls. After you select a specific location, click the "Impulse Responses" tab. Seek an impulse response (IR) audio file that is mono, i.e., single channel. Download one or more of the available IR audio files.

► Download the LabVIEW VI Resample Audio .wav available at `http://decibel.ni.com/content/docs/DOC-23105`. Use this application to produce a set of speech and IR audio files that have the same sampling frequency. Choose a standard sampling frequency such as 44.1 kHz.

2.6 Convolution reverb audio processor

Reverberation is a mass of echoes that cannot be individually perceived as discrete echos. Begin your design by creating a simple IR that contains distinct echoes for testing purposes.

► Study the video tutorial `http://youtu.be/ChSVf9lZ44k` to learn how to read and write .wav audio files, how to extract the audio file as a 1-D array for processing, and how to listen to the audio within the VI.

$\square_6$ Create a LabVIEW VI that generates an IR .wav file containing four unit values (impulses) separated by silence (zero values) at intervals 50, 500, and 900 ms. This is, the signal array looks like a 1 at $n = 0$ followed by 50 ms of silence, and then another 1, and then 500 ms of silence, and so

on. Remember to calculate the number of zeros in the silent regions using the same sampling frequency that you used for your speech and IR files. MathScript or a combination of Initialize Array and Replace Array Subset works well for this part. NOTE: Type `Ctrl+Space` to search for LabVIEW subVIs by name.

$\square_7$ Create a LabVIEW VI to read your speech and IR .wav files, convolve them, and then write the processed audio to a file.

- Remove the Convolution "algorithm" constant to revert back to the default method "frequency domain."
- IMPORTANT: Place the Quickscale subVI in the "Signal Processing | Signal Operations" subpalette just after the convolution subVI to ensure the audio signal does not exceed the ± 1 signal limits which would result in audible clipping noise.
- Add waveform graphs to plot the speech, IR, and processed audio arrays.
- Include three instances of the Play Waveform Express subVI to listen to each audio signal; connect the output error cluster of the first subVI to the input error cluster of the next subVI to guarantee the correct order of audio playback as speech, the IR, and then the processed audio.

$\square_8$ Test your convolution audio processor with the speech audio and the four-echo IR you created earlier. Discuss your test results. Which of the by-hand convolution problems above looks most like the echo IR?

$\square_9$ Now use the IR you downloaded from Open Air to process your speech signal. Discuss the degree of realism achieved by your convolution reverb audio processor.

$\square_{10}$ OPTIONAL: Once you have your reverb audio processor operational, you may enjoy creating additional sound effects:

- Reverse reverb – Reverse the IR signal array with Reverse 1-D Array in the "Programming | Array" subpalette.

- Non-reverb-based IR – *Any* audio signal can serve as an IR, not just those obtained by measuring a room. Try a short audio clip such as a speech utterance or a sound effect.