Lab 4: Artificial Reverberation and Room Simulation

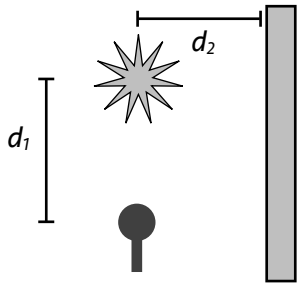
In this lab we will get some experience in simulating rooms digitally. We will design some simple reverbs using small filters and then work towards a more principled room simulator. You can use whichever sound you like for the following examples, usually a sound with sharp onsets is good for such experiments. One good example is this sound:

https://courses.engr.illinois.edu/cs498ps3/sounds/loop3.wav

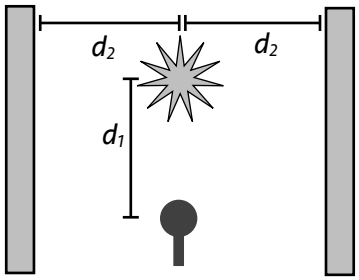
Part 1. Designing simple reverbs using filters

The simplest forms of reverbs can be designed using simple delays, comb and allpass filters. We will design one of each to get started.

a) A room with a single wall. For this reverb we will assume that we have a room that only has one reflective wall as shown below. Design a filter that simulates what the microphone will record. figure out the right filter in the case where d_i = [0.1, 1] meters, and in the case where d_i = [1, 10] meters. Let me know what the differences between these sounds are. To keep things simple, round the delays to an integer number and make up an approximate gain loss due to propagation and the wall bounce.



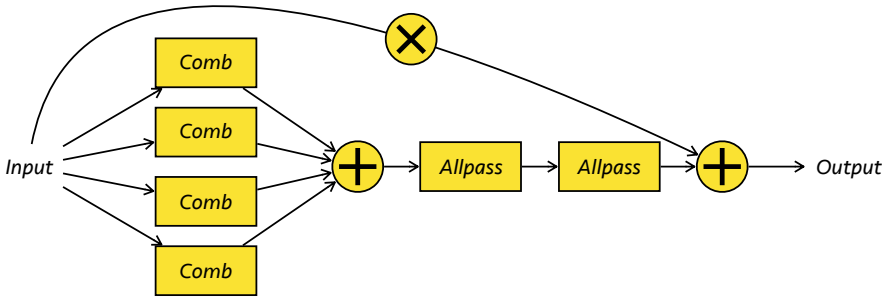
b) Now we will use a room geometry as shown below. Design the proper filter again, for d_i = [0.1, 1] meters and then for d_i = [1, 10] meters. Comment on the audible differences between these two settings.



c) Since for small values of d the output above gets its spectrum altered, we will use an allpass filter to make a more natural sounding echo pattern. Use the formulation shown in the lecture slides and compare that filter's frequency response with the response from the filter in the section b) above.

Part 2. Schroeder reverberators

Now we will combine multiple filters to make a more serious sounding reverb. Implement the structure shown below:



Try to find the necessary parameters for the above structure that makes it sound as good as possible (not too busy, not too subtle). *Hint*: There is no right answer! Try a few things, see what sounds best and explain why you used the parameters you decided to use. As you come across some bad sounding cases, describe what the problem was.

Part 3. Applying a real room response

To apply a more realistic filter we need to use a real room response. Download the St. Andrew's church room impulse response (Stereo) from:

http://www.openairlib.net/auralizationdb/content/st-andrews-church

Make sure that the sound you will convolve with it will be at the same sampling rate. This impulse response captures that church's RIR using two channels. To synthesize a sound in that room you need to convolve it with the impulse response. If your test sound is a single-channel recording, you can simply convolve it with each of the two impulse responses and then use the resulting outputs as the left and right channel. You will notice that using the `lfilter` command for this operation can take very long to compute. For faster convolutions we need to perform this filtering in the frequency domain. The function `convolve` automatically does that (if you use an up to date version), otherwise you can directly use `scipy.signal.fftconvolve`. Verify that the resulting output sounds like it has been placed inside a church.

Part 4 (extra credit). Measuring a room response.

Measuring a room response is crucial when trying to emulate a real room. In this exercise we will make a simple room response measurement. We will use the technique shown in the lecture slides.

a) First you need to generate a Maximum Length Sequence using `scipy.signal.max_len_seq`

b) Generate a sequence of 13 bits, this will output a 8191 point sequence (this is $x[t]$ from slide 36).

c) Go to a (preferably quiet) room that you want to measure, play the generated sequence from your computer and record the outcome (the easiest way to do that is to save this sequence as a wave file and play it while you record a new sound file). You can instead use the real-time I/O code from Lab 0. In doing so we will obtain the sequence $y[t]$ from slide 36.

d) Now you will need to perform the actual deconvolution. Take $x[t]$ and $y[t]$, compute their DFTs to get $X[\omega]$ and $Y[\omega]$, and estimate $H[\omega]$. Take the inverse DFT of H to get the time domain representation of the room impulse response.

e) Convolve the resulting response with the test sound above (make sure you use the same sample rates). Does it sound reverberated? This won't be perfect. What sounds "wrong" with it? Can you speculate why?