Assignment 5: Plucked string and reverberation

TA: Stefania Serafin
CCRMA*
serafin@ccrma.stanford.edu

In this assignment you will learn how to use very simple filters to build a plucked string physical model and put it in a reverberating environment.

1 Implementing a basic Karplus Strong algorithm.

Let's start by implementing a basic plucked string. This means that you will implement the following difference equation:

$$y[n] = x[n] + \frac{y[n-N] + y[n-(N+1)]}{2} \tag{1}$$

where $N = f_s/f_0$ is the length of the delay line, f_s is the sampling rate and f_0 is the fundamental frequency of the string. Note that f_s/f_0 in general is not an integer.

Write a Matlab function that does that. You function should look as follows:

```
y = kastro(freq,len, fs)
% freq = fundamental frequency of the string (in Hz)
% len = duration of the resulting soundfile (in samples)
% fs =sample rate (in Hz)
% y = vector containing the output samples.
% Note that the length of your delay line is N = fs/freq. Since this can be
% a real number, make it an integer by using round(N).
```

Inside your function you should implement a noise burst excitation which is the input to your plucked string, e.g. `(rand(1,N)-1/2)*Amplitude`.

2 Testing the plucked string

Due to the approximation made with the integer delay line, your plucked string is not perfectly tuned.

1. At a sample rate of 44100 Hz and a fundamental frequency of 330 Hz, what would be the length of the delay line in samples? What is the delay introduced by the low-pass filter in the delay line feedback path?
2. Try your function with a delay line of length N as found in the previous question. What fundamental frequency do you actually get?
3. Is the detuning more evident at high or low frequencies? Why?

3 Play your string in a room

Now your goal is to create some room effects around your string. As shown in class, Schroeder proposed two topologies for digital reverberation which are shown in figure 1 and 2 respectively. In it, AP represents an allpass filter while C represents a comb filter.

*http://www-ccrma.stanford.edu

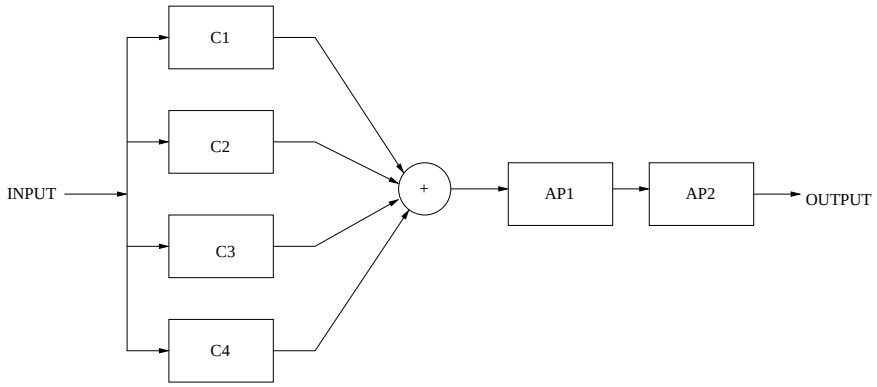

Figure 1: First reverberation topology.

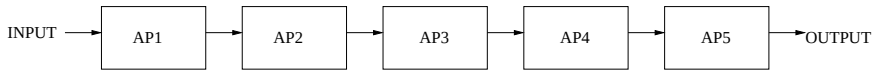

Figure 2: Second reverberation topology.

1. Implement in Matlab the two structures. For the first structure (the parallel feedback comb-filters C1 through C4 in Figure 1), try setting the delay time to 29.7 ms, 37.1 ms, 41.1 ms, 43.7 ms, respectively, and for the allpass filters, set the internal delay line lengths to 96.83 ms and 32.92 ms, respectively, and the decay times to 5 ms and 1.7 ms, respectively. Allpass decay time t_{60} is related to the allpass filters' coefficient by $t_{60} \approx 7T/[1 - g^{1/N}]$, where N is the allpass filter's delay line length in samples, and g is the coefficient magnitude. [Extra credit: derive this formula ;-)]

   Try to input to your reverberator a noise burst, like the one you implemented as input to the plucked string. Listen to the behavior of your reverberator to the noise burst.

2. Now try to find your own parameters, just remembering that prime numbers give less repetitious reverberation, and the allpass filters' coefficients should be less than 1 in order to have a stable filter.

3. Use the structure to reverberate the plucked string. This means that the output of your plucked string should be the input of your reverberator.

4. Listen to the string before and after being reverberated. Start listening to the non-reverberated string, then try with small delay lines (which would simulate a small room), and increase the size of the delay lines to see if this gives the sensation of a bigger room.

5. When you find some parameters that satisfy you, provide your implementation of the reverberating structures together with the parameters you used.

4 Plucked string in STK

For this part of the assignment you don't need to turn in anything; it is just if you want to get to know real-time plucked string synthesis in C++ using STK.

From your home directory, type the following:

```
cp -r /user/s/serafin/pub/Plucked .
cd Plucked
make
plucktst Mandolin -r < scores/funicula.ski
gedit READMEPlucked.txt &
```

Read what the README file says and play with it.