

# A Bowed String Instrument Modeling in Max/MSP

- *A MUMT 307 Final Project*

Boyu Deng  
260656496

## **Abstract:**

Commercially, digital musical instrumental approach to string instruments are mostly sampling-based, and some other times physical-modeling-based. Synthesized musical instruments often cannot mimic the real-life counterpart, but instead used as signature electronic musical timbres. This project proposes a method of using Schroeder all-pass filters as small reverbs which gives rise to a promising result to improve the palpability of synthesized string sound.

## **Objectives:**

This project features a two-part design of a purely synthesized bowed string sound based in Max/MSP. In the first part, a synthesis scheme for bowed string is achieved using only basic waveforms and amplitude modulation techniques; in the second part, digital delay lines are implemented as filters in [gen~] objects in Max/MSP to add space and colour.

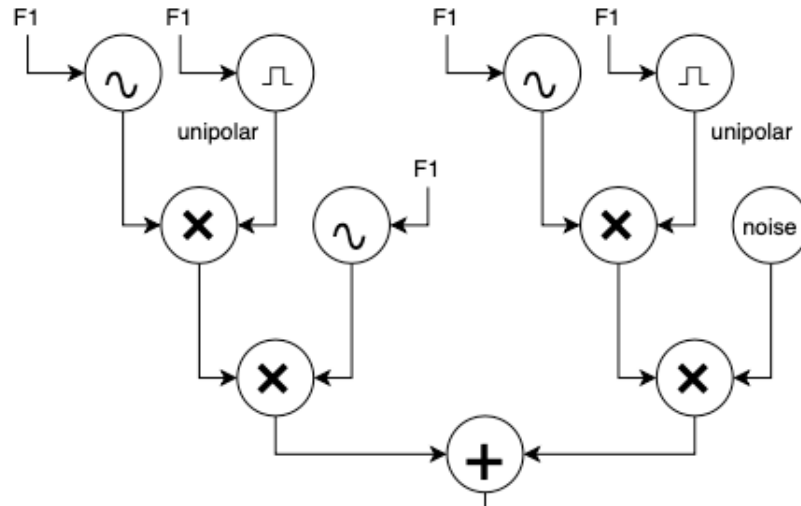
## **Methodology:**

### **I. Synthesis of bowing the string**

Max/MSP is not the best platform to do actual physical modeling because of its limitations in feedback mechanism, even within its lower-level sample-based processing object [gen~]. However, this project dedicates in using basic waveforms and amplitude modulation to implement characteristic properties of bow-string interaction and motion. The use of synthesis method instead of sampling string instrument yields many applied advantages: low disk-space occupation, more flexibility in modeling different bow and string materials.

There are many interesting phenomena exhibited by bow-string interaction, as discussed in (McIntyre, Schumacher, & Woodhouse, 1983). Due to the friction force applied by the bow onto the string, the catch-and-release interaction between them results in a sudden jump at some point of bowing, hence a minor skip for each effective oscillation period. This phenomenon is applied in this project by using a unipolar square wave with a duty cycle of less than 50% to amplitude-modulate another waveform; the use of a small duty cycle square wave AM almost results in an impulse-like train, and it is used to simulate the microscopic jumps that occur during bowing.

Additionally, this amplitude modulation is applied to not only a sinusoidal tonal component of the string sound, but also to a low-pass noise generator to add some scratchiness to the bowing sound, shown in **Fig. 1**. Stepwise synthesis examples are appended in **Appendix 1**.



**Figure 1. Synthesis of string bowing with amplitude modulations**

On the left side: a sinusoid signal is amplitude modulated by a unipolar square wave generator with a duty cycle of 0.27, this unipolar signal is then used to amplitude modulate another sinusoid signal, these three generators use the same frequency parameter. On the right side: same thing happens but the unipolar square wave generator has a duty cycle of 0.03, which eventually results in an impulse-like noise signal that is pitched. The two components (tonal, scratchy/noisy) are summed together

## II. Instrument body design

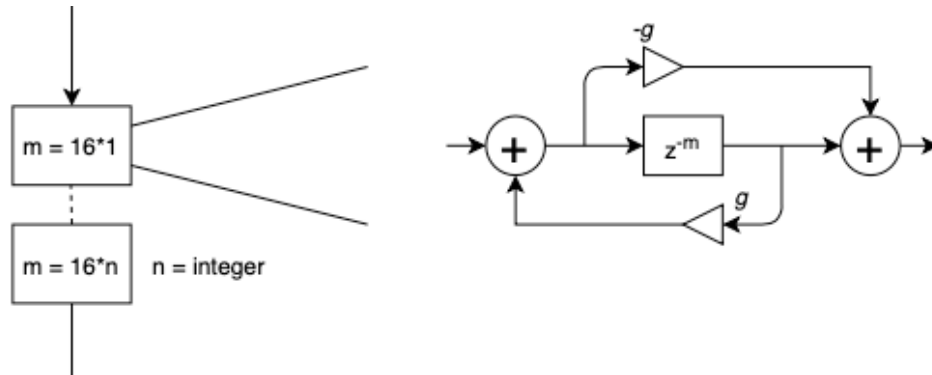
(Schroeder all pass filters as space-adding)

(part 2 resulting sound – add g,z tables for 2.1, add spectrum figures for 2.2, go to appendix for comparison.wav)

The instrument body design comes in two subparts: all-pass section and resonance section. The raw outcome from synthesis is quite dry and robotic, these two parts adds space and colour to the sound.

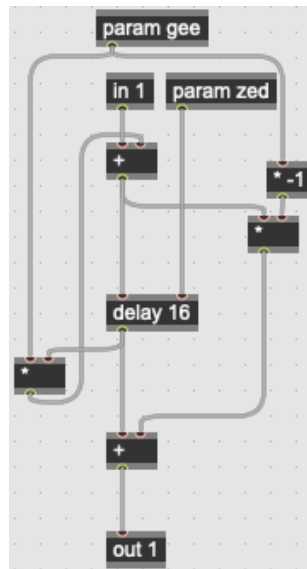
The first subpart consists of seven Schroeder all-pass filters. In theory, all-pass filters have flat frequency response but can alter phase response, and in this case, all-pass filters are used to mimic the small space that the instrument body has. 7 Schroeder all-pass filters are put in series. Instead of having mutually prime delay numbers ( $z^m$ ) as in the design on many late reverb schemes such as JCRev, this cascade of Schroeder all-pass section uses integer multiples of a small delay number. The hypothesis that undergoes the development of this project is that, since an instrument is magnitude smaller than a room, it would make sense to use delay numbers that share the same multiplier. **Fig. 2** shows the block diagram of this subpart, **Fig. 3** shows the

implementation of the Schroeder all-pass filter in [gen~] in Max/MSP, and **Table 1** displays the parameters of the Schroeder all-pass cascade.



**Figure 2. Schroeder all-pass cascade for mimicking an instrument body**

The all-pass filter cascade is consisted of 7 all-pass filters, each having the structure shown on the right side. The delay numbers are integer multiples of 16 samples, which corresponds to approximately 12.5-centimetre distance travelled at a sample rate of 44100Hz and 344m/s speed of sound in the hollow space inside the instrument body.



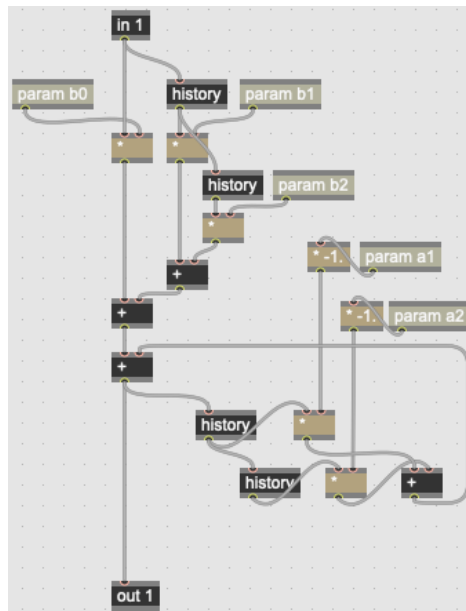
**Figure 3. Schroeder all-pass filter implemented in [gen~] subpatcher in Max/MSP**

Each all-pass filter is implemented identically. Parameters such as feedback/feedforward constant 'g' and delay number 'z' are specified for each instance through [param gee] and [param zed] shown in the subpatch.

**Table 1. Schroeder all-pass filter cascade parameters**

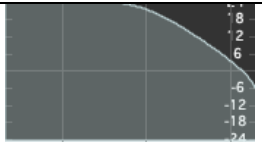


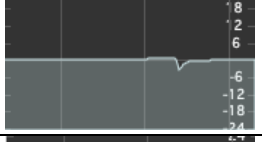
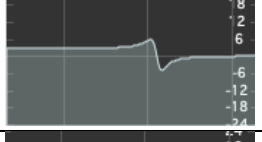

All-pass Filter #	Delay Number Multiplier	Delay Number ( $m$ ) (samples)	Feedback/Feedforward Constant ( $g$ )
1	1	16	0.79203
2	2	32	0.927368
3	3	48	0.716842
4	5	80	0.908571
5	7	112	0.85218
6	9	144	0.720602
7	11	176	0.85594

The second subpart is a cascade of 6 second-order resonance filters in series. This section of filters add a wooden colour to the sound, the parametric values are provided by (Maestre, Cook, & Scavone, 1995-2019). **Fig. 4** shows the implementation of the second-order filters in [gen~], this implementation functions identical to the built-in [biquad~] object but it was interesting to recreate using the [gen~] subpatcher. **Table 2** lists the filter coefficients provided by the STK class [bowed.cpp] and displays the frequency response of the 6 filters.



**Figure 4. Second-order filter implemented in [gen~] subpatcher in Max/MSP**

**Table 2. Second-order filter cascade parameters (Maestre et al., 1995-2019)**

Filter #	Coefficient b0	Coefficient b1	Coefficient b2	Coefficient a1	Coefficient a2	Frequency Response
1	1.0	1.5667	0.3133	-0.4409	-0.3925	
2	1.0	-1.9537	0.9542	-1.6357	0.8697	
3	1.0	-1.6683	0.8853	-1.7674	0.8735	
4	1.0	-1.8585	0.9653	-1.8498	0.9516	
5	1.0	-1.9299	0.9621	-1.9354	0.9590	
6	1.0	-1.9800	0.9888	-1.9867	0.9923	

**Result:**

The outcome from synthesis of bowed string alone is not significantly pleasant to the ear, the resulting sound was dry and deadly consistent, but it was an interesting combining the tonal and the scratch component together for a semi-realistic string sound. (appdx. 1)

In the finalizing of this project, the dry/wet technique is used in second part of sound polishing, where 85% the raw string sound is passed through all-pass section and then 65% of that is applied with the resonance filters. (appdx.2) In the appendix, there are also some examples showing what the sound may be like with and without the two filter cascades.

The result of this project is not the best string sound, but it is not too shabby for a totally synthesized sound with just basic waveforms. In the future if this model were to receive any improvement, I would start by exploring some actual physical modeling techniques, and experiment on more all-pass filter schemes and combinations to achieve an even more realistic sounding instrument body.

## Appendix:

### 1. Stepwise synthesis examples:

[synth\_tonal.wav]

The resulting synthesis of the tonal component of the string, as shown on the left side of figure 1.

[synth\_scratch.wav]

The resulting synthesis of scratch component of the string, as shown on the right side of figure 1.

[raw\_string.wav]

The resulting summation of tonal and scratch component of the string, shown in figure 1.

### 2. Filter cascade comparison examples:

[ap\_only.wav]

Raw string sound with only all-pass section applied.

[reson\_only.wav]

Raw string sound with only second-order filter section applied.

[final\_string.wav]

The combination of those two filter cascades enabled, and a dry/wet value of 85% and 65% orderly.

[final\_example.wav]

An example midi file played through this virtual string instrument using an ADSR envelope.

**Reference:**

Maestre, E., Cook, P. R., & Scavone, G. P. (1995-2019). bowed.cpp. *Synthesis Tool Kit*.

McIntyre, M. E., Schumacher, R. T., & Woodhouse, J. (1983). On the oscillations of musical instruments. *The Journal of the Acoustical Society of America*, 74(5), 1325-1345.