WoodGrains

Riccardo Alfieri, Riccardo Iaccarino, Manuele Montrasio,
Silvio Sgotto, Francesco Veronesi.

Group Name: *AKGigi* - Group ID: *10*

March/April 2022

1 Introduction

In the following document we discuss how we created in SuperCollider *foley* sounds by means of granular synthesis and developed a combined Graphical User Interface (GUI) that allows the user to manipulate some of the parameters of the sounds. *Foley* sounds are sounds that are usually created to be added to movies or videos, and thus they are sounds that belong to a certain environment. Hence, the software that we developed tries to recreate the hearing sensation of a specific environment.

Inspired by the city we study in, which is known as the city of luthiers, we chose to implement the sounds that belong to the luthier's lab. In particular we chose to recreate, using granular synthesis, the sounds of the following four objects:

- the rasp - see for reference Figure 1;
- the sandpaper - see for reference Figure 2;
- the hammer - see for reference Figure 3;
- the violin string (when plucked) - see for reference Figure 4.

Moreover, considering that a sound produced by granular synthesis is slightly affected by the track from which the sample is taken, we chose - as source file - a cello track. This choice somehow reverses the cause-effects relationship. Taking into account that the cello is one of the final products of the luthier, in this case the sound of the instrument becomes the starting point of the work. For our project we decided to work on an excerpt of the famous *Cello Suite No. 1* in G Major by J. S. Bach.

The report is organised in different sections. The first one presents, in general, how the "problem" of granular synthesis has been addressed. The second section, instead, outlines the implementation for the chosen *foley* sounds. In the last section, we focus on the development of the Graphical User Interface (GUI).

Figure 1: The rasp. Video frame taken from the video [2].

Figure 2: The sandpaper. Video frame taken from the video [2].

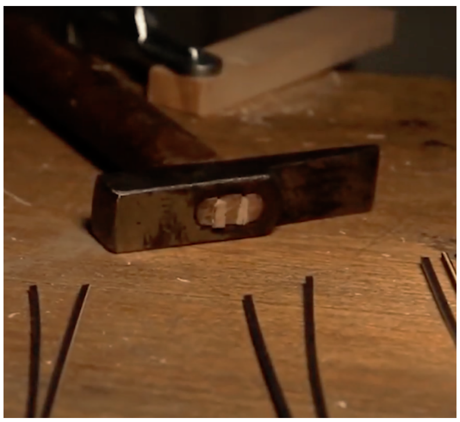

Figure 3: The hammer. Video frame taken from the video [2].

Figure 4: The violing strings. Video frame taken from the video [2].

2 Analysis of the problem

The main problem of this assignment was understanding how to produce a desired sound using granular synthesis. Granular synthesis is achieved by generating hundreds or thousands of very short grains - a grain is a *msec* signal with an amplitude envelope similar to a quasi-Gaussian bell curve - with the purpose to create a larger musical event. This technique can be classified as a sort of additive synthesis, in fact the resulting sound (the event) is obtained by adding together a large number of grains. Once the grains are organized and arranged into an event, it is possible to vary the global parameters to create a more "uniform texture". Granular synthesis offers an unique way to create a connection between the micro and the macro level. In fact, the micro-grain is a trivial and insignificant sound object, but - by means of an higher controlled level structure - it enables the generation on the macro level of a significant sound.

The primary control elements of granular synthesis are the overall amplitude of the texture, the grain density (how the grains are distributed within the event) and the inter-onset time (duration between two adjacent grains). SuperCollider (SC) provides a class called `GrainBuf` that allows to implement granular synthesis (from a loaded audio file) and to control the previous parameters - and also others.

The SC `GrainBuf` class contains the following arguments:

```
GrainBuf.ar(numChannels: 1, trigger: 0, dur: 1, sndbuf, rate: 1, pos: 0,
    interp: 2, pan: 0, envbufnum: -1, maxGrains: 512, mul: 1, add: 0)
```

We found that those that produce the most significant effects for the goals set (luthier's lab) are the `trigger`, `dur`, `rate` and `pos`. For this reason we decided to modify only these parameters and leave the others with their *default* value. The

- `trigger` determines how densely the grains are distributed within the texture. It affects the timbre, the amplitude and the pitch of the produced sound. To produce noises it is useful to use an irregular trigger frequency, for example with the class `Dust`. Instead, to produce sounds it is effective to use a regular trigger frequency with the class `Impulse`.
- `dur` defines the time duration of the individual grain. It is interesting to observe that when the duration of the grains is greater than their playback period, an overlap between adjacent grains occurs. If the duration is long enough, the source track can be recognised.
- `rate` modifies the playback frequency of the grains. It affects the pitch and the timbre.
- `pos` defines the point on the source track where the granulation starts. It can modify the timbre either more or less slightly, depending on the type of sound produced and the type of source track.

The sound produced by each object of the `GrainBuf` class was further shaped by an envelope and different effects.

To better recreate the "luthier lab" environment, which is composed by different and repetitive sounds, and so to achieve the feeling of a real situation we decided to loop each sound. In order to improve the reliability of the *foley* sounds created - a succession of sounds that are all the same is unnatural - we include a random component in some of the sound parameters. For example for the waiting time between one sound and the next one.

3 Implementation of the *foley* sounds

Each of the sounds created - each of which will be discussed individually in this section - has been obtained from the same audio file: an extract, as mentioned in the introduction, from the *Cello Suite No. 1* by J. S. Bach.

To make the sounds as close as possible to those of reality, we *(i)* modelled them using as a reference those audible and recorded in the video [2] (the same from which the Figures 1, 2, 3, 4 where taken) and *(ii)* added a reverb to the sounds to enhance the spatial details linked to the room where the luthier works.

Moreover, for each sound implemented a correspondent *UGen Graph* has been realised. To save space, we include the corresponding images only at the end of the report.

3.1 The rasp sound

The rasp produces a sound with a complex spectrum of high and low frequencies. Hence, we decided to build the synthesised signal by appropriately combining three different elements, which differ mainly in `rate`.

The way in which the luthier uses the rasp is very irregular. Despite this, the sound that the tool produces on a piece of wood has mainly two fundamentals features: its amplitude and its "mainly heard" frequency increase over time. These aspects, in the granular synthesis structure, can be controlled with the trigger frequency and by means of an appropriate envelope. The latter, shown in Figure 5, has been implemented following a linear ramp. Its attack needs to be set relatively long with respect to the release, which is way shorter.

The roughness of the sound can be simulated exploiting a `Dust` *UGen*, which makes the frequency of the granular pulses more random. The other non-deterministic variables used to model the sound are the rate of the highest pitched element, the envelope attack and the time between two consecutive sounds in the loop, which ensure the final result to be even more realistic.

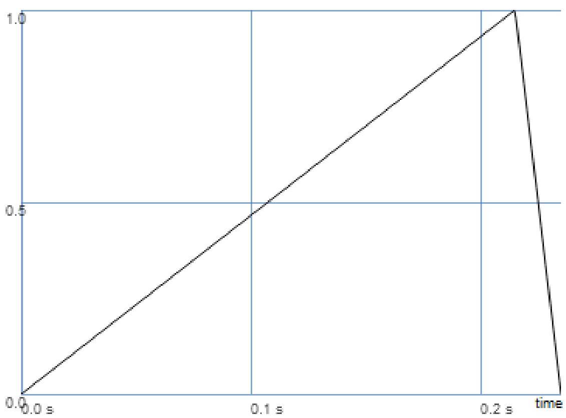

Figure 5: The envelope used for the rasp sound.

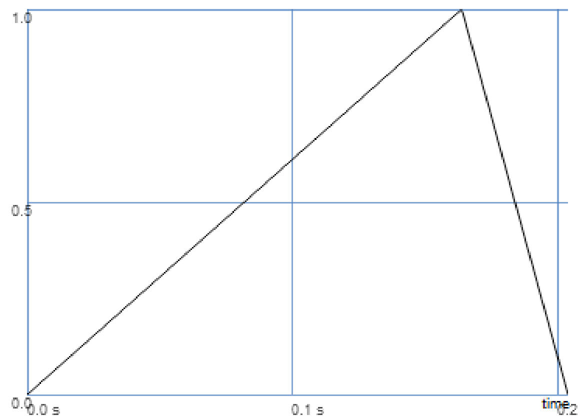

Figure 6: The envelope used for the sandpaper sound.

3.2 The sandpaper sound

The sandpaper rubbing sound on wood is mainly composed of high frequency components. Hence, we chose the following values for the principal parameters:

- `trigger = Dust.ar(5000)`. The frequency of the trigger is very high and it is irregular thanks to the properties of the class `Dust`. This configuration ensures a high pitch and a "noisy" sound.
- `dur = 0.0001`. To avoid the superposition of the grains `dur` must have a very low value.
- `rate = 40.midiratio`. This value (approximately 10.08) has been chosen to be in accordance with `trigger`.

The envelope of the sandpaper is quite similar to the one of the rasp, as can be seen in Figure 6, because it has to reproduce the same type of movement, which has an increasing velocity that causes a gradual increasing volume of the sound.

3.3 The hammer sound

The hammer sound is mainly percussive, a feature that can be achieved only by means of a proper envelope. The one that has been implemented, shown in Figure 7, has a very short attack and a much longer release. The hammer sound, dynamically controlled by the presented envelope, is composed by superimposing three different signals which differ from each other in their `rate`, with the purpose of creating a sound with an approximately uniform spectrum. Despite this, the reverb has been applied differently on each of the synths that compose the sound: we chose to emphasise by means of the effect (the reverb) primarily the low frequencies of the sound.

Lastly, for the `trigger` we found more effective to use the class `Impulse`, which guarantees a less rough sound.

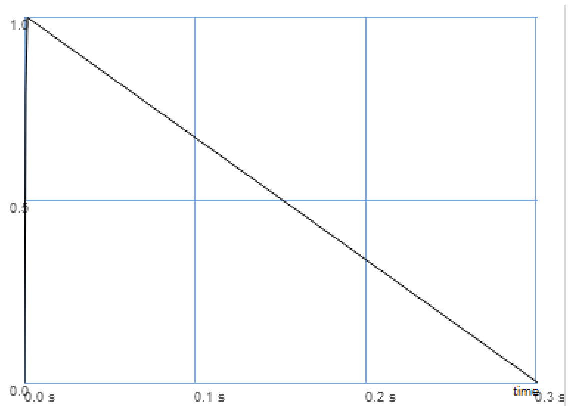

Figure 7: The envelope used for the hammer sound.

3.4 The plucked violin string sound

Among the four synthesized sounds, the plucked violin string certainly is the most musical and tuneful one. For this reason, also in this case an `Impulse` *UGen* has been chosen to control the trigger's frequency and thus the pitch of the plucked string, helping the signal to achieve the characteristic smoothness of the instrument. Moreover, we have investigated the paper [1] in order to analyse the spectral composition of the plucked violin strings, which happens to be a weighted sum of three different harmonics: band-pass filters helped us in this task.

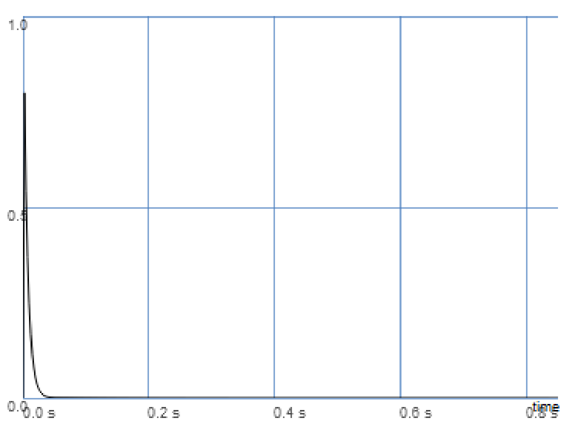

Figure 8: The first envelope used for the plucked violin string.

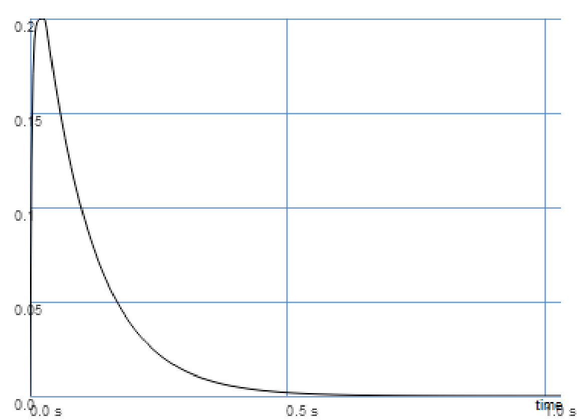

Figure 9: The second envelope used for the plucked violin string.

Regarding the choice of the envelopes, two different shapes are involved in this Synth (`Env.perc` and `Env.adsr`). The former (whose shape is visible in Figure 8) is responsible for recreating the action of the plucking finger, whereas the latter (see for reference Figure 9) is related to the actual sound generated by the vibration of the string itself.

Lastly, different parameters have been designed to be dynamically changed by the user, such as the tremolo effect, the timbre and the pitch (the note of the string) of the synthesized sound.

4 Implementation of the Graphical User Interface

The Graphic User Interface (GUI) that has been implemented for the assignment is shown in Figure (10). In the top-left of the window it is possible to find a button (1) that reproduces the source track (the *Cello Suite No. 1* by J. S. Bach). The rest of GUI is divided in four boxes, one for each of the implemented sound. They are all structured in the same way, as follow. In the top there are two buttons, the first one (2) allows to play the single *foley* sound and the second one (3) starts a loop in which the sound is repeated until it is clicked a second time. Immediately under them, there are two sliders to control volume (4) and pan (5), respectively. The square in the centre of the column (6) is the temporal plot of the sound of the left and the right channel. With the button (7) the user can change the domain of representation of the sound. Finally, in the bottom part it is possible to find two rows of knobs. In the first row (8) the user can change some of the parameters of the granular synthesis and of the envelope, depending on the sound. In particular, from left to right, for the

- `Rasp`: the curve coefficient of the envelope, `rate` and `pos`.
- `Sandpaper`: the `trigger` frequency, the duration of the grain and the curve coefficient of the envelope.
- `Hammer`: `rate`, the duration of the grain and `pos`.
- `Pizzicato`: the note, the frequency of the tremolo and `pos`.

In the second row (9) it is possible to change the parameters of the reverb.

5 Conclusion

In general, we would like to say that only by means of a deeper understanding of granular synthesis, obtainable through more experience and more time, it would have been possible to create more accurate sounds.

In particular, at the beginning of the work it was not immediate to understand how to modify the parameters of granular synthesis in order to achieve specific variations of the sound timbres. For this reason we found difficult to choose how many and which sounds to recreate and to decide which parameters the user could modify without changing radically the properties of a sound.

Despite this we are satisfied with the final result both from the conceptual point of view - the idea of the luthier's lab sounds - and also from the "audio" point of view - the similarity between the sounds reproduced by us and the real ones.

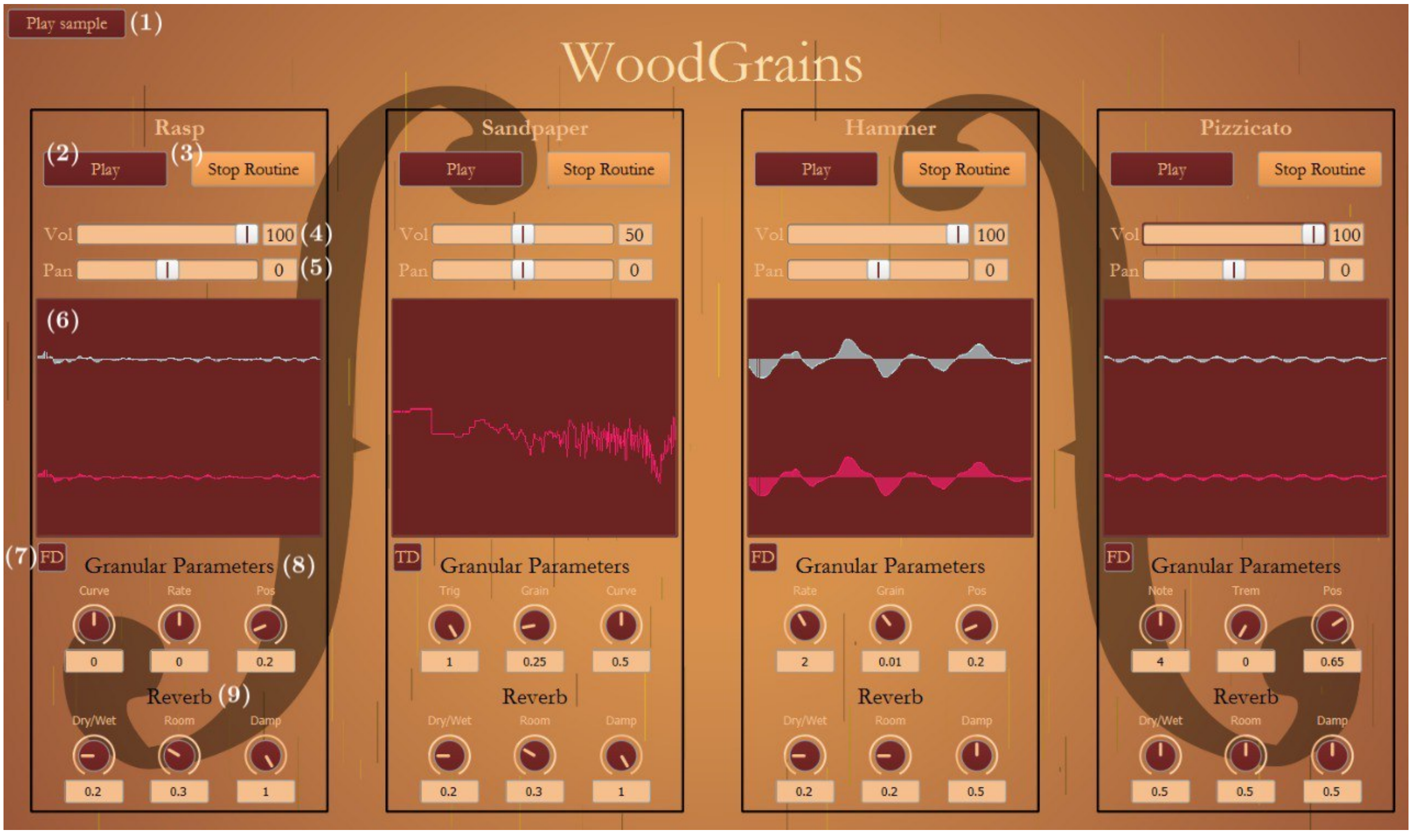

Figure 10: The Graphic User Interface (GUI) of the project. The button (1) plays the original sample, (2) plays the *foley* sound once and (3) plays the *foley* sound multiple times. The slider (4) changes the volume and (5) pans the *foley* sound. The `scope` window (6). The Button (7) changes the domain of representation of the sound. The knobs (8) control general timbre properties of the sound and (9) change some reverb parameters.

References

[1] Rachel Rebagay. *Frequency analysis of bowing and pizzicato as a function of placement on a viola a string*. URL: https://repository.tcu.edu/bitstream/handle/116099117/10416/Frequency_Analysis_of_Bowing_and_Pizzicato_as_a_Function_of_Placement_on_a_Viola_A_String.pdf?sequence=1. (accessed: 15.03.2022).

[2] *Unintentional ASMR – Violin Making (Carving, Scraping Wood, Rough Sounds)*. URL: https://youtu.be/YO5KMZfyKBc. (accessed: 15.03.2022).

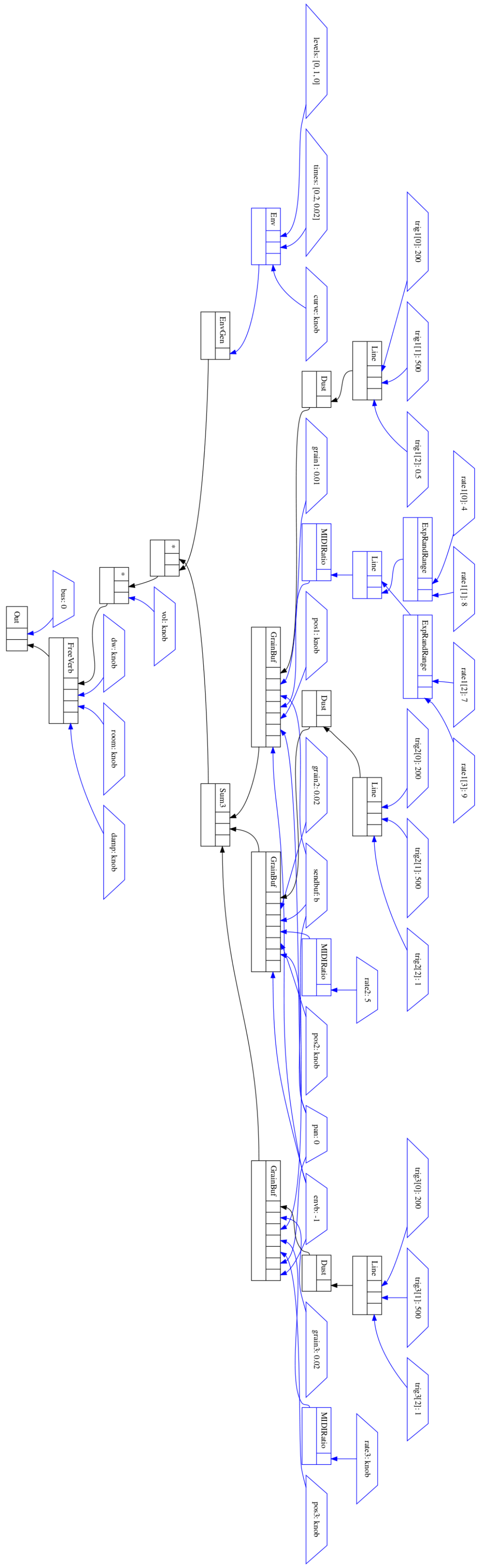

Figure 11: The *UGen Graph* of the rasp synth.

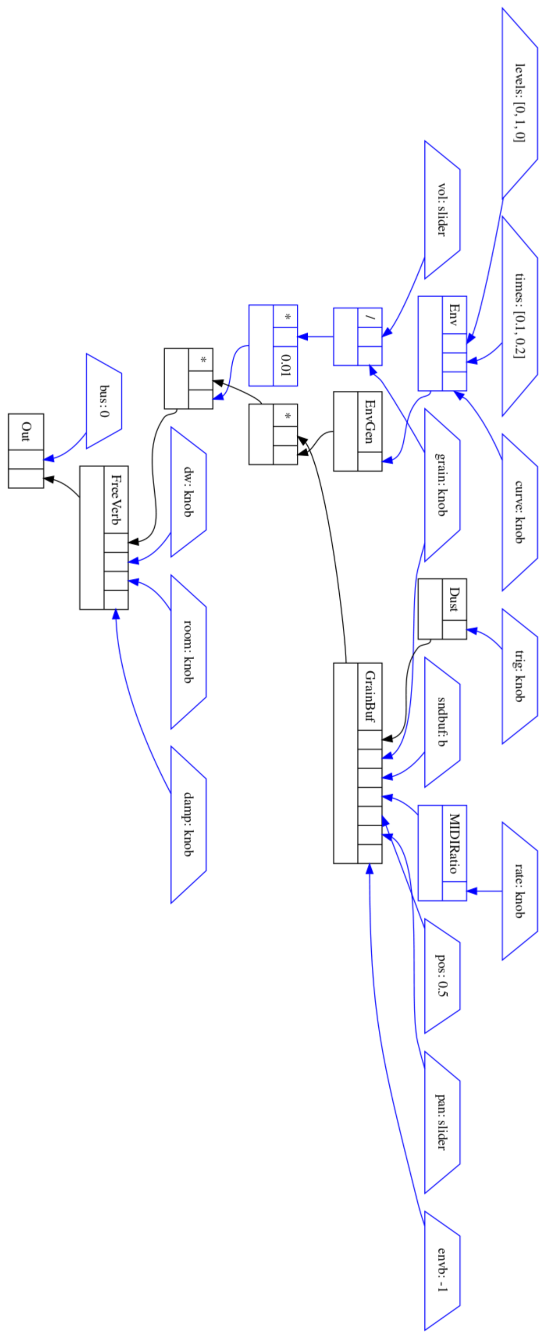

Figure 12: The *UGen Graph* of the sandpaper synth.

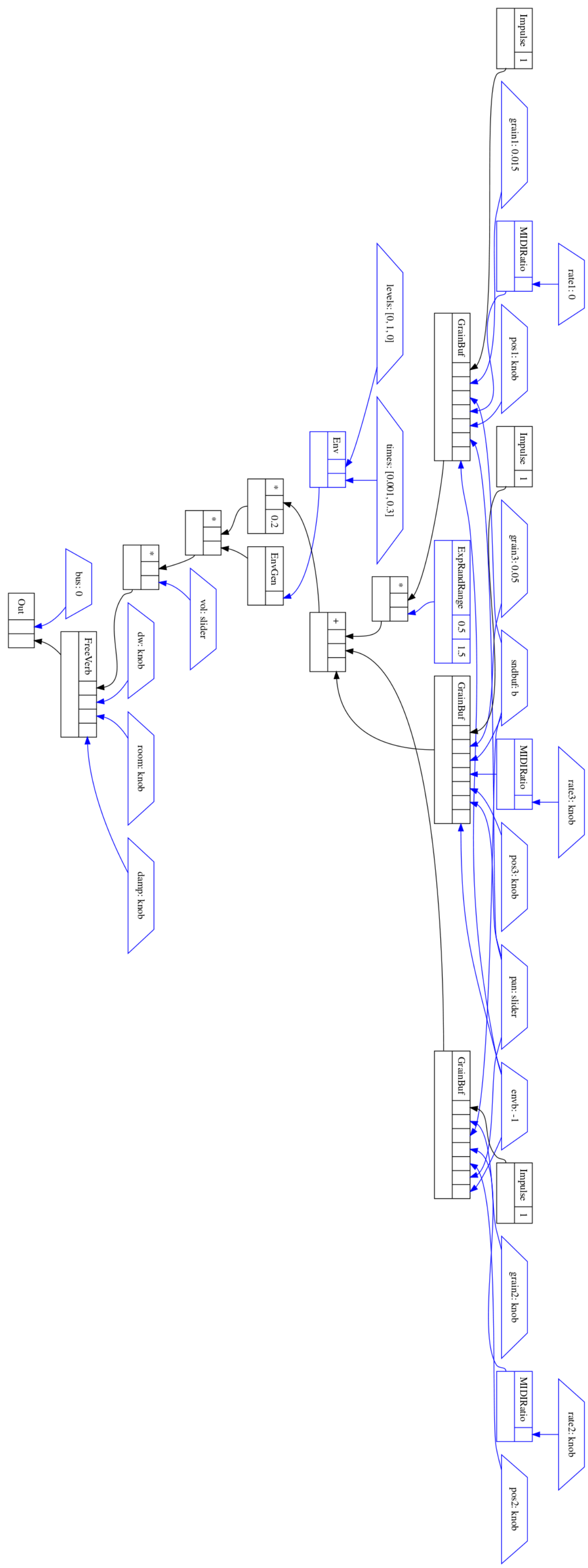

Figure 13: The *UGen Graph* of the hammer synth.

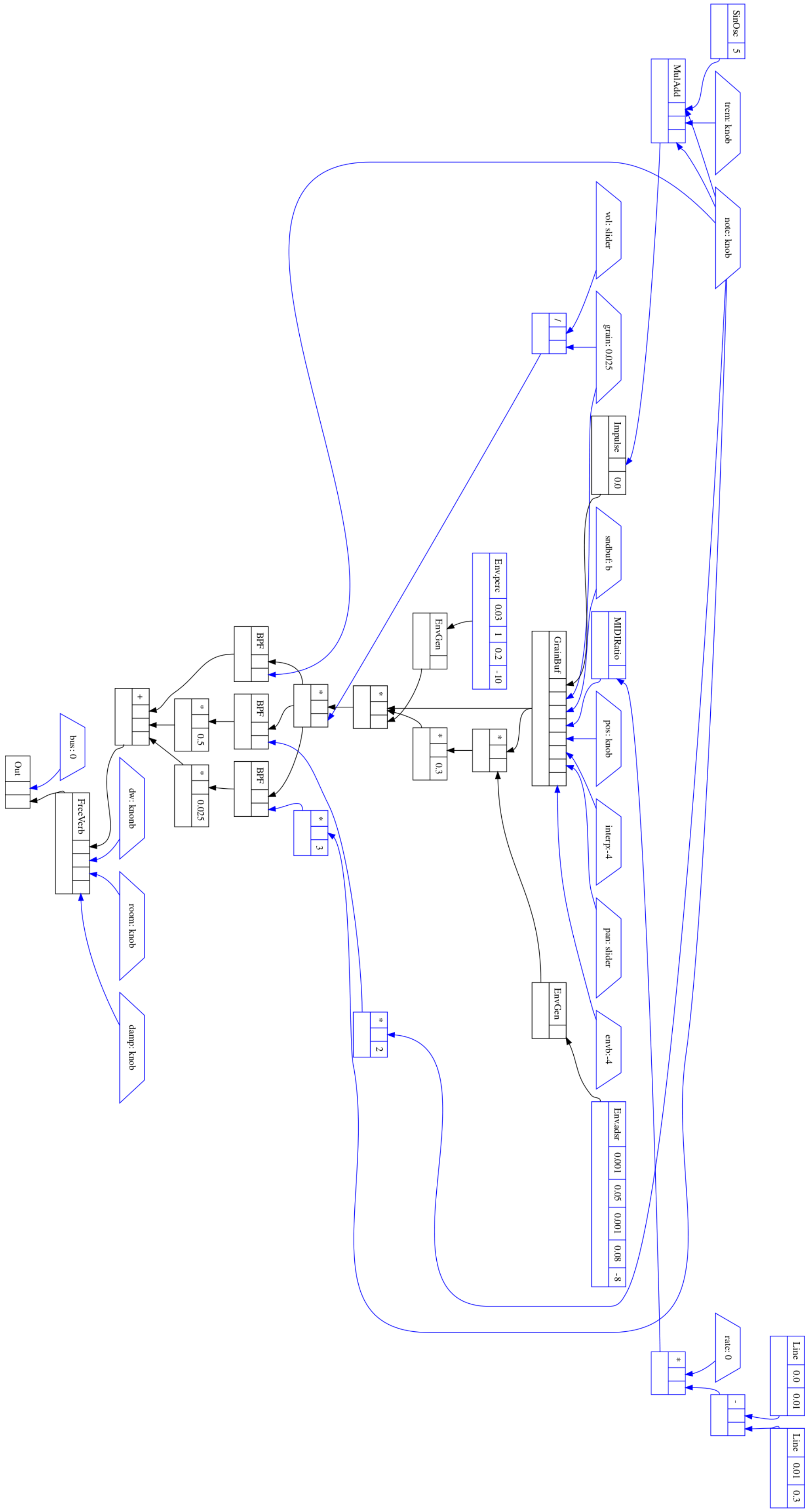

Figure 14: The *UGen Graph* of the plucked violin strings synth.