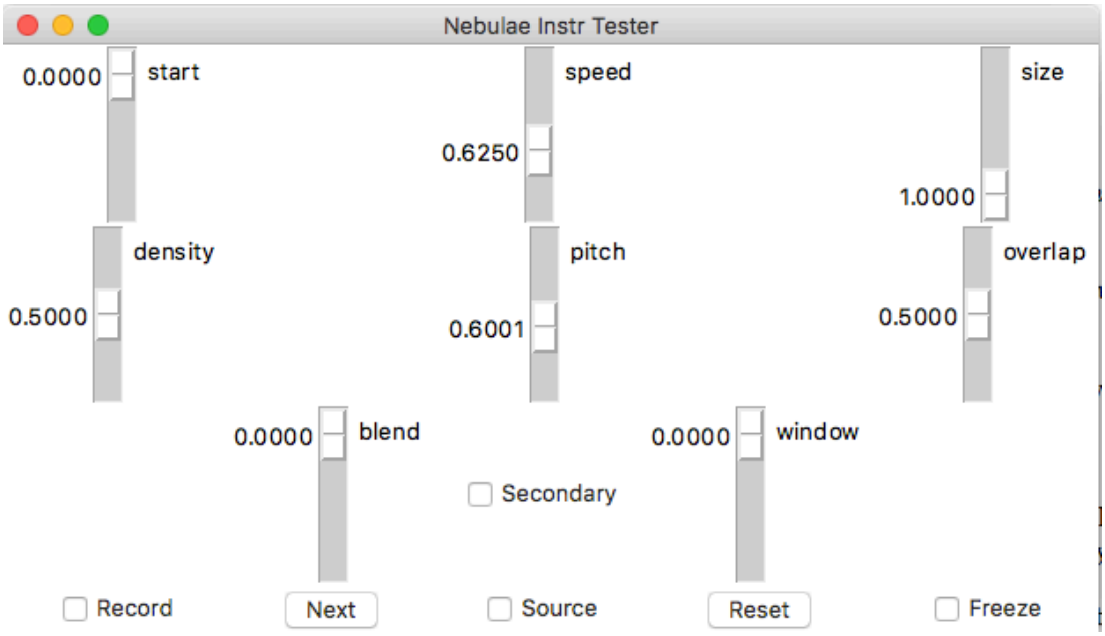
Instr-Tester



Contents

Instr Teseter

This is a cross platform Nebulae Emulator running the same platform that runs on the actual module with a GUI generated by TKinter.

Currently the Instr Tester runs on Mac and Linux.

Windows support as well as an executable/packaged form should be available in the near future.

Requirements

Csound (>= 6.0.9 has been tested)

Python 2.7

Numpy (sometimes part of scipy package)

TKinter (should be included in Python install)

Usage

```
# Navigate to package directory
cd path/to/Nebulae_Instr_Tester
# Run the tester:
python instr_tester/nebulae.py <instr name - no extension>
# for example:
python instr_tester/nebulae.py a_granularlooper
```

The instr tester references files from a few locations within the Nebulae_Instr_Tester folder.

./audio - load in audio files, mono tables of first channel in each file will be loaded for use by the default methods. See the instr-file-reference for details on using Stereo files. *This simulates where audio files end up on the hardware of the nebulae*

./instr - load in .instr files here. I've included several to get started, and explore. Typing the name of the file with no extension as the argument to the python program will run the tester with that instrument.

./matrices - these are specifically for scanned synthesis, and can be used. Not all of them have been tested, and some work with the scanx opcodes, while others work with the standard scanned opcodes.

./log - the instr tester will output a copy of the csound score followed by the orchestra as a file called formattedcsd.txt - This was initially meant for debugging, but I've left it in place since it may be useful.

You'll notice there is an additional "Secondary" button. This flips all of the controls over to their secondary parameter. The sliders will update to the current position for the control mode.

One thing worth noting, the pitch control is calibrated on the hardware so that 1V of CV is equivalent to 0.2 difference in control.

Limitations

Buttons

Currently there is no support for handling the buttons differently. In one of the next few updates I plan on changing which type of buttons are used based on the nebconfig section of the .instr files.

Because of this, there may be some weird behavior with instruments where the buttons are reconfigured (like b_oneshot.instr)

The fixed state of the buttons for now work as follows:

Button	Behavior
Record	Latching Checkbox
Next	Incrementing Button
Source	Latching Checkbox
Reset	Timed message (50ms pulse of the value 1 to gkreset)
Freeze	Latching Checkbox

Because of this, the alternate behavior for the buttons has not been implemented yet, either.

Audio I/O

For now there is limited control over the I/O. Csound automatically uses the default adc/dac.

So setting up your input/outputs can be done from your OS.

I'll eventually add configuration for this, especially so audio could be routed through a standard DAW.

I've only tested audio input on a Macbook Pro using the built-in microphone.

Defaults

The standard defaults from defaultnebsettings.txt are loaded correctly, however overwritten defaults set within the nebconfig section of the .instr file are not recalled properly.

This will be fixed in the next update.

Plans for expansion

These lists are just rough ideas, and are not guaranteed to be implemented

A few things that will be updated in the near-term:

- Dynamic Button handling based on nebconfig of selected instrument.
- File load/Instr load options for manually selecting a file.
- Configurable ADC/DAC selection
- Buttons for clicking faders back to original (to simulate hardware behavior)

And on a slower timetable:

- Add the ability to name controls so that they overwrite the value
- Add the ability to test pd patches designed for the nebulae (very easy, just need to setup the socket)
- Replace the generic TKinter UI with something a little more true to the module so that it actually looks like the hardware.