sound_source_id

Release 0.2.6

Samuele Carcagno

May 06, 2024

CONTENTS:

CHAPTER

ONE

SOUND_SOURCE_ID

`sound_source_id` is a program for testing static sound localization. The interface is shown in Figure *Screenshot of the sound_source_id interface.*.

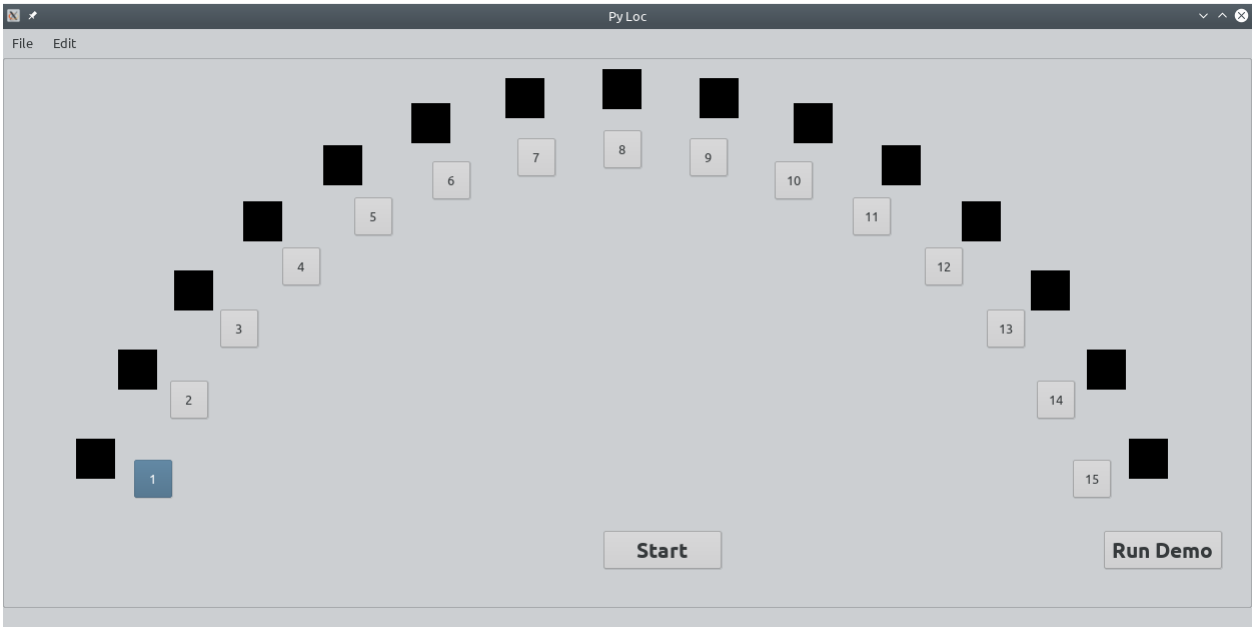

Fig. 1: Screenshot of the `sound_source_id` interface.

CHAPTER

TWO

INSTALLATION

`sound_source_id` has been successfully installed and used on Linux and Windows. It should also work on Mac platforms, but this has not been tested. `sound_source_id` is written in Python and can be installed via pip:

```
pip install sound_source_id
```

`sound_source_id` depends on a few Python modules including:

- PyQt6
- numpy
- scipy
- matplotlib
- pandas
- PyAudio https://pypi.org/project/PyAudio/

depending on your Python distribution you may want to install these dependecies before installing `sound_source_id` via pip (e.g. through conda if you're using the Anaconda Python distribution or through your Linux distribution package manager if you're using the Python installation that comes with your Linux distribution), otherwise pip will attempt to automatically pull in and install these dependencies. If the program is successfully installed you should be able to start it from a bash/DOS terminal with the command:

```
sound_source_id
```

You need to ensure that the Python environment you're using when you call the above command matches the one you used when you installed the application.

Sound can be played with either PyAudio, or SoX on Windows. On Linux pyalsaaudio can be also used. Depending on how you want sound to be played, you need to install:

- pyalsaaudio https://pypi.org/project/pyalsaaudio/

or SoX:

- https://sox.sourceforge.net/

CHAPTER

THREE

PARAMETERS FILE

The settings for a test are stored in a parameters file, which is a plain text file. An example parameters file is shown below:

```
angles : -70, -60, -50, -40, -30, -20, -10, 0, 10, 20, 30, 40, 50, 60, 70
labels : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15
channels : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15
n_chan : 15
n_blocks : 1
stim_list_file : stim_list.csv
randomize : true
demo_stim : pink_noises/noise1.wav
demo_stim_lev : 65
```

The following fields need to be specified in a parameters file:

- *angles*: the azimuth angles (in degrees) at which the sounds are presented. Note that a 0° angle indicates straight ahead, a 90° angle is to the right, and a -90° angle to the left.
- *labels*: a label for each of the angles, this can be a number or a letter (e.g., a, b, c, ecc.)
- *channels*: the channel of the soundcard that will be used to present a sound at the corresponding angle
- *n_chan*: the total number of channels for the setup
- *n_blocks*: the number of blocks, that is how many times the test will be repeated
- *stim_list_file*: the path (absolute or relative) to the file containing the stimulation list (see below)
- *randomize*: if *true* the stim_list_file will be shuffled before each block repetition
- *demo_stim*: the path to the WAV file to be used for the demo
- *demo_stim_lev*: the sound level (in dB SPL) to be used for the demo

CHAPTER

FOUR

STIMULATION FILE

Stimulation files specify the stimuli that will be played on each trial of the test. Figure *Example stimulation file.* shows an example stimulation file.

	A	B	C	D	E	F	G
1	angle	sound_file	condition	level	roving	feedback	
2	-70	pink_noises/noise1.wav		65	6	true	
3	-60	pink_noises/noise2.wav		65	6	true	
4	-50	pink_noises/noise3.wav		65	6	true	
5	-40	pink_noises/noise4.wav		65	6	true	
6	-30	pink_noises/noise5.wav		65	6	true	
7	-20	pink_noises/noise6.wav		65	6	true	

Fig. 1: Example stimulation file.

Each row of the file represents a trial. Stimulation files contain the following columns:

- *angle*: the angle at which the sound will be presented (stimuli will be sent to the corresponding soundcard channel as specified in the parameters file)
- *sound_file*: the path (relative or absolute) to the WAV file to be played
- *condition*: an optional label specifying the experimental condition
- *level*: the *base* sound level (in dB SPL) at which the sound will be presented (this assumes that *sound_source_id* has been correctly calibrated)
- *roving*: a level rove, actual sound level will be euqual to the base level plus a value drawn from a random uniform distribution between +/- the roving level
- *feedback*: if *true*, feedback will be given to the listener at the end of each trial

CHAPTER

FIVE

SOUND LEVEL CALIBRATION

Figure *Edit transducers dialog* shows a screenshot of the `Transducers` dialog which is used for setting calibration values.

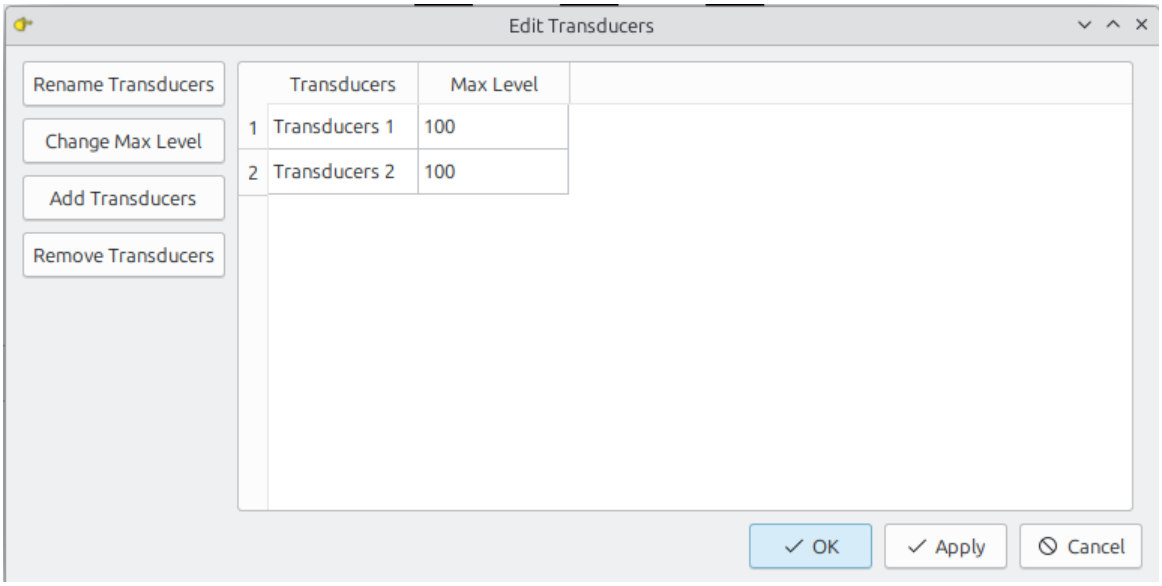

Fig. 1: Edit transducers dialog

Most of the fields should be pretty much self explanatory. Using this dialog you can add headphones/speakers models to the transducers database. In *sound_source_id* levels are referenced to the level that would be output by a full amplitude sinusoid (a sinusoid with a peak amplitude of 1). In the `Max Level` field you should enter the level in dB SPL that is output by the transducer for a full amplitude sinusoid . However, getting reliable readings for a pure tone with an SPL meter is difficult, therefore, typically a noise is used for calibrating loudspeakers.

The procedure I normally use for calibrating loudspeakers is to save on disk a noise stimulus as a wav file. I filter the noise within the operating range of the SPL meter (usually around 0.05 to 8 kHz). The noise level needs to be reasonably high as to avoid signal-to-noise ratio issues, but not so high as to cause distortions or damage your hearing in the measurement process. Once I've found a reasonable level, by trial and error, I measure the actual level with an SPL meter held at the position where the listener head would be located relative to the loudspeaker during the experiment, and note it down.

We can measure the root-mean-square (RMS) level of the WAV file with the noise used for calibration, let's call it RMS_{noise}. A full amplitude sinusoid has an RMS amplitude of $1/\sqrt{2} = 0.707$. The difference in dB between the

level of a sinusoid at max amplitude and our calibration noise will be equal to:

$$\Delta_{dB} = 20 \log_{10} \left(\frac{1/\sqrt{2}}{RMS_{noise}} \right)$$

Therefore, if our calibration noise had a level (measured with the SPL meter) of x dB SPL, a sinusoid at max amplitude would have a level of:

$$maxlev = x + \Delta_{dB}$$

this is the value that you need to enter in the `Max Level` field of the transducers calibration table for the loudspeakers in question.

CHAPTER

SIX

INDICES AND TABLES

- genindex
- modindex
- search