Audio Communication Group
2024

Documentation of Pure Data Applications for Dynamic Binaural Simulations

First draft: 10.07.2024
Latest Updated: 28.07.2024

Author:
Konstantin Fontaine — k.fontaine@tu-berlin.de

Supervision:
Emanuele Porcinai — emanuele.porcinai@tu-berlin.de
Fabian Brinkmann — fabian.brinkmann@tu-berlin.de

Contents

1 General Information

The following documentation presents the components and use-cases of a Pure Data (PD) implementation for dynamic binaural simulations for single or multiple persons. The chapters cover a presentation of the core components, binding setup and naming requirements and a brief explanation of demo main applications. For the simulation, a partitioned convolution of live inputs is implemented, based on the SoundScape Renderer (SSR) PD externals[1], as well as a dynamic binaural playback of pre-convolved source materials, both dynamically adaptive to the users head orientation. At the current state, the implementation solely provides dynamic rendering for horizontal head movements. **Most components are implemented as abstractions and can be replicated as indexed objects within projects.**

2 Components

2.1 Source Input

The *source_input* abstraction serves as interface to handle the amplitudes of incoming source signals. The first four interface input channels are assigned automatically as depicted in the GUI. A master gain serves to control the overall amplitude of incoming signals. To amend the input channel numbers, open the subpatch and edit its *dac* ~ object, which contains the corresponding interface channel numbers or simply add an offset in the audio setting.

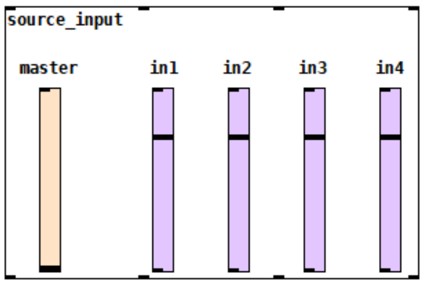

Figure 1: Source Input Mixer

2.2 Source Player

For the playback of mono files, the *source_player* provides one of four audio files to be played through its outlet. The GUI hereby offers options to start

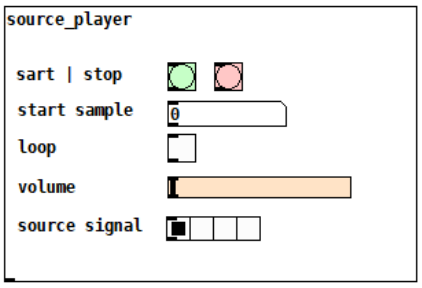

Figure 2: Sample Player

[1]https://ssr.readthedocs.io/en/0.6.1/use-cases.html

playback from a determined sample, loop the file or stop playback on demand. The four samples (drums, speech, guitar, noise) are automatically loaded from the disk when running the main patch and can be selected from the GUI. To customize files, open the subpatch and adjust the fileloader strings.

2.3 File Loader

For the dynamic binaural playback of convolved sound files, the *file_loader* abstraction must be setup accordingly. Hereto angular resolution (e.g. 2° steps) and bipolar range (e.g. 60 for -60° to 60° coverage) must be defined. Once set, a file container is generated containing a pre-convolved representation of the sound field. To assure its correct file management, refer to chaper 3.2.

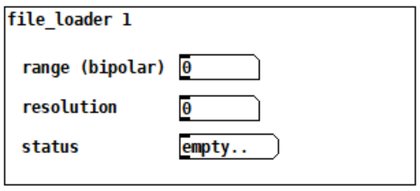

Figure 3: File Loader

2.4 OSC Receiver

Tracking data is handled within the *osc_receiver* abstraction, which offers OSC port selection and activation toggle, to allow data transmission within the main application. Head orientation is depicted by yaw, pitch and roll and incoming OSC messages must have the shape ”string1 string2 string3 <yaw> <pitch> <roll>”, while the strings are discarded during processing.

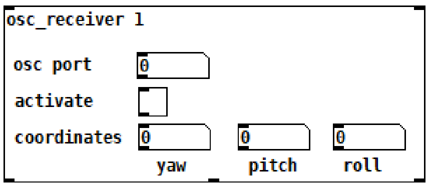

Figure 4: OSC Receiver

2.5 Binaural Convolver

Dynamic convolution of multiple sources is realized in the *convolver* abstraction, which automatically receives OSC data and source inputs. The four input sources of the live mixer are assigned to the four inlets of the convolver in ascending order and can be muted for testing purposes. Within that abstraction two *ssr_brs* ∼ objects[2] convolve each incoming signal with the according BRIR set, in either a dynamic or static manner. While the dynamic part aligns the filter with corresponding head orientation, the static part convolves the unaltered frontal head orientation. For the latter, preceding zero-padding is required for the duration of the dynamic component since both renderers obtain incoming signals simultaneously. To assure its correct file management, which is bound to file name and folder structure, refer to chaper 3.1. For replication, its abstraction is defined for 1 to 4 sources, whereby the first argument always depicts the index while the second argument defines the audio threads of each renderer.

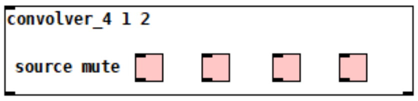

Figure 5: Binaural Convolver

2.6 Binaural Player

In case of pre-convolved sound fields, the dynamic playback is realized in the *binaural_player* abstraction. Dynamically changing head orientations pick the according reference convolution from the file container described in section 2.3. Furthermore a start sample can be selected, the playback be looped and the volume be adjusted. Additionally the desired sample overlap depicts the blend time in samples. A turquoise signal feedback at the bottom indicates incoming OSC data. For its use a file loader and OSC receiver with the same index are required.

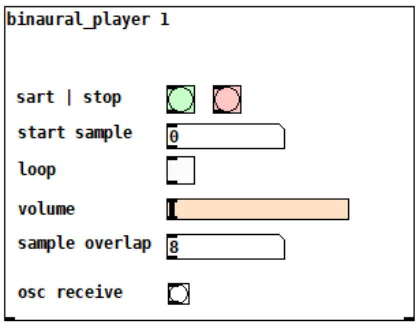

Figure 6: Binaural Player

[2]https://ssr.readthedocs.io/en/latest/use-cases.html

3 Folder and Data Structure

In the following those requirements are described for the *convolver* and the *binaural_player* to assure their correct file management. In general, indices below 10 are represented by single digit numbers within filenames.

3.1 Binaural Convolver

The BRIR sets for the dynamic convolution are automatically read from disk and assigned to the renderers, once the patch is started or when a new instance of the abstraction is added to a project. The folder structure in which the sets are located should therefore look as depicted in figure 7.

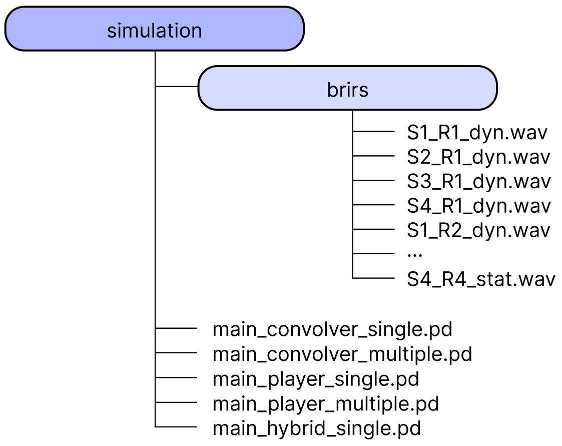

Figure 7: Folder Structure for the BRIR Simulation

To pick and assign the BRIR sets automatically, all file names must contain source index, receiver index and an indicator for the static (*stat*) or dynamic (*dyn*) BRIR component, all separated by underscores as depicted in the figure (e.g. *S1_R2_dyn.wav* for the dynamic part of the second receiver with source index one). For the case of four simultaneously performing musicians, the brir folder should thus contain 32 wav files covering all receiver-source-combinations in dynamic and static manner. Each dynamic BRIR set must then contain 720 channels that are constructed as presented in the SSR binaural room simulation renderer documentation.[3]

[3]https://ssr.readthedocs.io/en/0.6.1/renderers.html

All BRIR sets must be provided in order to successfully instantiate the renderers according to their initial source count. In case the entire BRIR should be dynamically convolved, the static files can be discarded, while the occurring initialization error at patch start can be neglected.

3.2 Binaural Player

For dynamically playing back already convolved sound streams, the *file-loader* subpatch requires files by the name convention $convolved\_Rx_1\_x_2azim\_x_3elev\_left.wav$, while x_1 denotes the receiver index, x_2 the yaw value and x_3 the pitch value followed by the corresponding ear signal (*left/right*). Once the range and resolution are set, the *binaural_player* automatically obtains the buffer indices and becomes available for dynamic playback.

4 Main Applications

Five demo main patches are provided in the simulation folder, for live convolution, preconvoled use or in combined manner. In case of the live convolution patches, DSP setup and activation is recommended prior starting the patches, to ensure a correct object initialization. The PD sampling rate must match with the sampling rate of the BRIR sets. Additionally main applications might benefit from a tracking data inversion which is provided in all demos.

4.1 main_convolver_single.pd (Single Player)

In the single player live convolver, an automatic assignment of four BRIR sets (each representing distinct source positions for one receiver) is applied when starting the patch. Usage thus only requires OSC port settings and activation. When input streams are muted or equal to zero, computation is compensated and thus more efficient. The four mixer inlets offer alternative source inputs to be passed through to the renderer, here exemplary the mono stream from the source player output can be convolved.

4.2 main_convolver_multiple.pd (Multiple Players)

The multiple player main application, enables four live convolutions of each four receiver-source-combinations. Each of them convolve the dynamic early component and a subsequent static part in alignment to the head orientation of the individual player. Tracking data is automatically committed to the corresponding convolvers once their ports are selected and activated. Simply add further convolvers in combination with OSC receivers of the same index.

4.3 main_player_single.pd (Single Player)

When dynamically simulating pre-convolved sound fields by the means of the main_player patch, the file container must be setup initially, as described in chapter 2.3. In a subsequent step OSC port selection and tracking activation then enable the simulation.

4.4 main_player_multiple.pd (Multiple Players)

For multiple players the same processing is applied. To assure coherence amongst the players, the read pointer positions and playback-related GUI parameters are kept in synchronization to one another, while the head orientation remains individualized for each musician.

4.5 main_hybrid_single.pd (Single Player)

The hybrid mode combines a dynamic convolution of four live inputs with a synchronized dynamic playback of pre-convolved files. To expand to multiple users simply add the same abstractions with a new index.