WONDER Configuration Files

cwonder

cwonder_config.xml

settings	
projectpath	This determines the directory in which cwonder will store its projects by default if no further arguments are provided with the OSC message "/project/save".
maxNoSources	The maximum number of virtual sound sources that will be rendered. This information is transmitted via OSC to all modules that connect to cwonder.
	This determines how many audio input channels will be provided by twonder.
renderpolygon	This determines the area inside which sound sources will be rendered as focused sources. It is made up of an arbitrary number of points (s.b.).
roomname	The name of the room. Currently only used by xwonder to display it in the user interface.
point	This represents one point of the renderpolygon. Please keep in mind that WONDER uses a right-handed coordinate system with the z-axis pointing UPWARDS in real space.
x	x coordinate, when looking down on the room, this is left(-)/right(+)
y	y coordinate, when looking down on the room, this is up(-)/down(+)
z	z coordinate, when looking down on the room, this is away(-)/towards(+)

WONDER Configuration Files

twonder

twonder_config.xml

focus

limit	Focussed sources that have a greater distance to the speakers than this limit will not be rendered. In meters.
margin	Currently NOT USED!

speakers

distance	This determines the transitional area around the speakers in which amplitude changes have to be compensated in order to allow an inaudible movement of sources through the speakers. In meters.
reference	This constitutes the reference distance a speaker uses for rendering. In meters. For the actual rendering algorithm, please refer to lines 99-150 in the sourcefile source.cpp of twonder.

twonder_speakerarray.xml

segment	This represents one segment of speakers. A segment can be an arbitrary number of speakers, but they have to be arranged in an equidistant, linear fashion. You may use multiple segments in this configuration file if one instance of twonder should address multiple segments of speakers.
id	Each segment must carry a unique ID. Just use ascending integer numbers starting with 1.
numspeak	The number of speakers that make up this segment. Minimum is 1.
winwidth	Currently NOT USED!

startx	The x coordinate of the beginning of the speakersegment.
starty	The y coordinate of the beginning of the speakersegment.
startz	The z coordinate of the beginning of the speakersegment.
endx	The x coordinate of the end of the speakersegment.
endy	The y coordinate of the end of the speakersegment.
endz	The z coordinate of the end of the speakersegment.
normalx	The x coordinate of the normal vector of this segment. The normal vector should be a unit vector.
normaly	The y coordinate of the normal vector of this segment. The normal vector should be a unit vector.
normaly	The z coordinate of the normal vector of this segment. The normal vector should be a unit vector.

WONDER Configuration Files

scoreplayer

scoreplayer_config.xml

settings

scorepath	This determines the directory in which the scoreplayer will store its scores by default if no further arguments are provided with the OSC message "/score/save".

WONDER Configuration Files

fwonder

*** = optional**

fwonder_config.xml

jack	The configurations regarding the interfacing with the JACK infrastructure.
name	The name under which fwonder should register with JACK.
number_of_sources	The number of audio inputs fwonder offers. You need a complete set if IRs for each input.
number_of_outputs	The number of audio outputs fwonder has. Must be at least 2. More outputs are valid, but currently NOT USED.
brir	Data about the binaural room impulse responses
path	The absolute path of the directory where the IRs are stored. The individual IRs for the source X have to be stored in the subdirectory /sourceX according to the following naming scheme: Example: N15_P360.wav for a (N)egative elevation of 15 degrees and a (P)ositive azimuth of 36.0 degrees. Azimuth uses one decimal digit, but that has to be 0.
azimuth	The horizontal (x-axis) range for which IRs are available.
start	Most negative IR position. In degrees.
stop	Most positive IR position. In degrees.
elevation	The vertical range (y-axis) for which IRs are available.
start	Most negative IR position. In degrees.
stop	Most positive IR position. In degrees.

impulseresponse

do_crossfades	This determines whether the results of the blockwise convolutions will be crossfaded with each other.
max_length	This sets the maximum allowed length of the IRs. If an IR is shorter it will be zero-padded, if it is longer, it will be truncated. In samples.

tail*

name	The name of the file that contains the IR that should be used as a tail, that is appended to all other IRs. Each source has exactly one tail located in the same directory as the other IRs.
max_length	This sets the maximum allowed length of the tailIR. If the IR is shorter it will be zero-padded, if it is longer, it will be truncated.
partition_size	Sets the size of the chunks of samples of the IR that should be processed in one block. Must be at least 1. In samples.
window	Possible values are LINEAR, NOWIN, COS2. CAUTION: this seems to affect ALL IRs not only the tail.
do_crossfades	This determines whether the results of the convolution with the tail-IR will be crossfaded with the results of the convolution with the source-IRs.
offset	This sets an offset of the results of the tail-IR convolution in respect to the source-IRs. So the tail-IR are delayed. In blocks of samples, depends on the JACK buffersize.

static_ir_matrix*

x_resolution	Resolution of the static IR matrix in the left/right direction (azimuth). In degrees.
y_resolution	Resolution of the static IR matrix in the above/below direction (elevation). In degrees.

dynamic_ir_matrix*

x_resolution	Resolution of the dynamic IR matric in the left/right direction (azimuth). In degrees.
y_resolution	Resolution of the dynamic IR matrix in the above/below direction (elevation). In degrees.
x_radius	Number of IRs that should be dynamically loaded to the right and left of the center IR.
y_radius	Number of IRs that should be dynamically loaded above and below the center IR.

advanced_settings*	These are settings for special purposes that are not part of the usual usage of fwonder. For experimental use.
resolution_changeable	Activates the feature that the resolution of the cache can be set via OSC. This automatically sets the static resolution to 1|1. This will not be visualized by qfwonder. This will override the settings for the dynamic cache resolution on each received OSC message.

Elevation[direction+degree]_Azimuth[direction+degree].wav

WONDER Configuration Files

tracker

tracker_config.xml

tracker	
type	This determines the type of tracker that is connected to your computer. Valid are ptracker and itracker
oscclient	This represents a client which is listening for OSC messages from the tracker. You may insert multiple clients into this file in order to send the tracker data to more than one client. Each client should use a different port when running on the same computer (i.e. same IP-address).
host	The IP-address the data should be sent to. Use "localhost" when the client is running on the same computer as the tracker.
port	The port the data should be sent to.
sendPan	1 = send panning data, 0 = no panning data will be send
sendTilt	1 = send tilting data, 0 = no tilting data will be send
sendRot	1 = send rotation data, 0 = no rotation data will be send NOTE: if all three kinds of data should be sent, the tracker will generate a /WONDER/tracker/move message, otherwise it will be a /WONDER/tracker/pan or tilt or rot message