The WFS system at La Casa del Suono, Parma

Fons Adriaensen
Casa della Musica, Parma

Linux Audio Conference 2010
Utrecht, Holland
La Casa del Suono

- A museum dedicated to the history of audio technology.
  * Created by the City of Parma.
  * A collection of vintage audio equipment.
  * A view on current and future technology.

- Two audio installations using Linux:
  * The 'Lampadario Acoustico'
  * The 'Sala Bianca'
• A room of approx. 7.5 by 4.5m, entirely white.

• 189 small speakers around the full inner perimeter, including the doors.

• All speakers driven individually, together they form a Wave Field Synthesis system.
La Sala Bianca – 2

- One speaker every 12cm.
  * Constructed in blocks of 10, 15, or 17.
  * Two-way bass-reflex with mixed order crossover.
  * Designed by Audio Link, using components from Ciare.
  * Very good performance for its size.
  * Frequency range is 50Hz to over 20kHz.
Based on the Kirchoff-Helmholtz integral:

The wave field inside a source-free volume $V$ delimited by a surface $S$ is completely determined by either the pressure or the volume velocity at all points of the surface $S$.

Practical realisation requires some compromises:

* Replacing the infinite number of points by a grid of discrete sources.
* Reduction from 3D to 2D.

WFS recreates the sound field of any number of primary (virtual) sound sources using an array of secondary (real) sources.

A variation of the technique can be used to create sources inside the volume or perimeter.
The Sala Bianca system will support up to 48 moving primary sources, rendered in real time.

Source movement is implemented by sample-rate update of rendering parameters, not by crossfading between static source positions. This means that e.g. the Doppler effect is rendered as well.

The system is used

- As part of the museum, for public demonstrations of the technology.
- For scientific research: as a listening room allowing virtual speaker setups, and for further development of WFS algorithms.
- As an instrument for electro-acoustic music concerts.

Planning started in 2007, and the system is in use since February 2009.
A primary source at $X = 4, Y = 3$. 
A primary source inside the perimeter.
The audio system – Hardware

- 4 Siemens/Fujitsu PCs running ArchLinux
- RME MADI interfaces, convertors and matrix.
- 192 channels of Aphex DA converters.
- 24 8-channel QSC amplifiers.
- Lots of cables.
The audio system – The control room

- Originally meant to be an Ambisonics room at double the size.
- Contains all PCs, (also for the Lampadario), network HW,...
- Mic preamps and analog audio lines to Sala Bianca, for recording, concerts, etc.
The audio system – Structure

DACs, amplifiers and speakers

MADI

ADAT

PC wfsren1

ADI648

MADI Matrix

PC wfsren2

ADI648

PC wfsren3

ADI648

DACs ADCs

MADI

ADAT

PC wfsmaster

ADI648

DACS, amplifiers and speakers
Using a Gigabit network for audio connections instead of MADI was considered but rejected.

- The synchronisation issues could be solved easily.
- Solutions were emerging, but untested for e.g. 48 channels.
- Performance of the hardware and drivers was not at all guaranteed.
- It would probably result in larger system latency.
- It would not permit major savings for the audio hardware.
Software architecture

- wfsmonitor: display and control
- wfsmaster: synchronisation
- wfsrender: signal processing

Connections:
- Levels and status from wfsmonitor to wfsmaster
- Timestamped positions from wfsmaster to wfsrender
- Audio timecode from wfsrender to wfsmaster
- Positions from wfsmaster to wfsrender
- Audio from wfsrender to wfsmaster
WFSmaster

- Command line application, controlled via WFSmonitor or Python supervisor.
- Provides a single access point for source position and movement commands.
- Translates these commands into periodic updates sent to the render applications.

- Ensures **audio and data synchronisation** in the entire system:
  * Source movement data must be applied with sample accuracy on all machines, but
  * jack periods on the render machines are not synchronised, and
  * position updates arrive asynchronously via the network.

* An audio time code generated by WFSmaster is used as a time reference by the render machines, and
* position updates are timestamped using this code.
WFSrender

- Command line application running on all rendering machines
- Controlled and monitored via WFSmonitor.
- Receives mono source signals and audio time code via the audio interface.
- Receives source position commands via the network
- Computes signals for up to 64 speakers.
- All instances are equal and read hostname to determine the set of speakers to use.

- Major parts are implemented as plugins:
  - The layout plugin defines system geometry.
  - The engine plugin defines the DSP algorithms.
WFSrender - Layout plugin

● Defines the geometry of the speaker array.

● Provides functions to read or calculate:
  * Speaker coordinates,
  * Vectors orthogonal to the line of speakers,
  * Distance of a point to the line of speakers,
  * Inside or outside determination of source locations,
  * Calibration data and parameters.

● Also allows linear (non-closed) arrays.
WFSrender - Engine plugin

• Contains two main routines:
  – Parameter calculation called by the network receiver thread.
  – DSP code called by jack_process().

• A lock-free buffer is required between the two.
The DSP code

• Control rate calculation of rendering parameters.
  – Each combination of (primary, secondary) source depends on a specific delay and filter.
  – The filter can be decomposed into two fixed filters combined with variable gain factors.
  – This results in a delay value and two gain factors for each combination of (input, output).
  – Control rate is variable, but normally set to 1024 samples (around 25ms or 40 updates per second).

• Audio rate calculation:
  – Two filters for each input.
  – Implementation of the sample-accurate synchronisation.
  – Linear interpolation of the three rendering parameters.
  – Calculation of the speaker driving signals.
  – An optional correction filter for each output.
  – Calculations can be optimised for stationary sources.
The basic algorithm

Three parameters for each (input, output) pair, updated at sample rate: \( \text{delay}, \text{gain}_1, \text{gain}_2 \).
External interfaces – 1

• Primary source position can be controlled via two interfaces:

  • **Shared memory interface.**
    - Available only on the *wfsctl* computer.
    - Allows sample-accurate synchronisation of audio and control data.
    - Used by LAPSPA plugins, WFSmixer and Python code.

  • **OSC interface.**
    - Available anywhere on the network.
    - Used by major composition and synthesis tools.
    - Basic commands allows controlled-velocity movement.

• Available audio interfaces are analog, ADAT and MADI.
External interfaces – 2

LADSPA plugin
Supercollider
Csound
Puredata
wfsmixer
Ardour
Python
Audio
wfsmaster
shared memory
OSC/UDP
OSC to renderers
Audio to renderers
dummy Jack connection
• Start, stop and monitoring of the master and render applications.

• Monitoring includes:
  – Program, network and timecode status.
  – Timing of rendering commands.
  – Primary source positions.
  – Speaker levels.

• Also provides test facilities, solo,...
• Original version (shown) provided level control, panning and simple movements.
• New version has also in-line panning, EQ, Aux sends and stereo monitor.
• ‘Total recall’ and OSC remote controlled.
Automatic operation

- Fully automatic operation under control of a Python supervisor program.
- Audio playback via PyJackPlayer, a multichannel player app implemented as a Python class.
- Most other components are Python classes as well.
- The supervisor program also acts as a server to one or more remote control clients running on EEE-PCs.
Many thanks to all who made it possible to realise this project:

- Prof. A. Farina, University of Parma.
- The president and direction of La Casa della Musica.
- My former colleagues at Audio Link.
- The authors of the Linux sound system, in particular Jack.
- All members of the LAD list who have provided help and hints.