SeamLess Integration of Spatial Sound Reproduction Methods

1
Henrik von Coler
TU Berlin
voncoler@tu-berlin.de

2
Paul Schuladen
TU Berlin
p.schuladen@tu-berlin.de

3
Nils Tonnätt
TU Berlin
n.tonnaett@tu-berlin.de

ABSTRACT

The presented software system aims at a combined use of different spatial sound reproduction methods, such as Wave Field Synthesis and Ambisonics, in a robust, user-friendly workflow. The rendering back-end is based on free and open source components, running on multiple Linux servers, thus allowing the operation of large loudspeaker setups. Using a send-based signal routing paradigm with an OSC message distribution software, the individual rendering engines can be combined seamlessly and extended with additional methods. Content can be created and played back using digital audio workstation projects which are unaware of the reproduction systems and make use of OSC automation plugins. This ensures a straightforward transfer of content between different sites and speaker configurations. Due to its adaptability, the proposed system is considered a potential solution for comparable setups with larger numbers of loudspeakers.

1. INTRODUCTION

Different methods for spatial sound reproduction, such as Wave Field Synthesis (WFS), Higher Order Ambisonics (HOA) or panning approaches like Vector Base Amplitude Panning (VBAP), have individual strengths and drawbacks. Depending on the requirements and resources, either can be preferable for a specific application. Multiple methods can also be combined to be used either alternating or in parallel. This results in versatile systems for varying or extended use cases with the ability of creating highly immersive sonic experiences.

1.1 TU Studio

For several years, the *TU Studio* at TU Berlin operates a sound field synthesis studio for music production and research. It is equipped with a 192 channel WFS system, a 21 channel loudspeaker dome for HOA and other methods, plus a classic ring of eight loudspeakers. With few exceptions, as for example in demos, these systems are usually used independently. One reason for this is the lacking availability of comparable systems and the high effort of installing them temporarily for concerts or shows. However, a recently built public listening room at the Humboldt Forum in Berlin offers new possibilities and requires an improved workflow for everyday operation. This paper introduces the *SeamLess* software system, using the TU Studio and the Humboldt Forum as example systems.

1.2 Humboldt Forum Listening Room

The Humboldt Forum is the new home of several museums and a cultural platform at the center of Berlin. Located in the reconstructed Berlin Palace, it encompasses amongst others the Ethnological Museum of Berlin, the Museum of Asian Art and the Humboldt-Lab. For immersive presentations of relevant audio content, the Ethnomusicology branch of the museum commissioned a multichannel system, combining a 2-dimensional WFS System and a 3-dimensional HOA System.

A previous project of the Audio Communication Group dealt with the planning, operation and evaluation of a multichannel loudspeaker system at the original location of the Ethnological Museum in Berlin Dahlem [1]. This predecessor was a 21 channel Ambisonics system. The software solution was based on VST plugins by Matthias Kronlachner [2] inside Reaper.

1.3 Requirements for the Combined System

The combination of different reproduction systems requires a new software concept, since the fully integrated DAW solution can not be extended to drive the WFS with the given number of channels. The result is a distributed software system, relying on task-specific components, with a special focus on the everyday use in exhibitions with low maintenance.

In the filed of computer music, respectively experimental electronic music and electroacoustic music, most composers and performers are used to individual tools and workflows. Especially for spatialization this offers the most powerful means when controlling a large number of sound source positions and other sound qualities. Despite powerful plugin based environments, spatial practice can still be considered a domain for technically skilled or experienced computer musicians. However, since the system at Humboldt Forum should be accessible to composers and artists from various backgrounds, a seamless operation as in usual DAW handling is aimed at. Further, the everyday use in a museum routine demands a server-capable, headless rendering system with a robust automation concept for sound source movements. All processes need to be started and organized in fully automated routines, in order to allow the operation by the museum staff and ensure synchronicity with other media processes.

Since the listening room in the museum is open to the public during daytime, it is necessary to prepare content in other studios. Projects thus need to be interchangeable, ignoring differences in the hardware configuration. These recurring problems are addressed by the SeamLess system.

1.4 Spatialization with WFS and HOA

Recent spatial audio systems usually work with the concept of virtual sound sources. This is also referred to as the object based [3] approach, in contrast to channel based approaches, which consider specific loudspeaker setups. In object based spatialization, audio signals are linked to dynamic spatial source positions. The actual loudspeaker signals are then generated in real time by applying these source positions within the rendering algorithms. With this concept, the audio material is detached from the presentation format and the content can be transferred to different loudspeaker setups or systems. This applies to Ambisonics, WFS and VPAB.

Figure 1 shows the model of a virtual point sound source with spherical coordinates, as used within the system. The position is defined by the angles azimuth (α) and elevation (ϵ) and a distance (d) from the origin, which is the center of the listening space. Cartesian coordinates, with the dimensions x, y and z can also be processed by the proposed system.

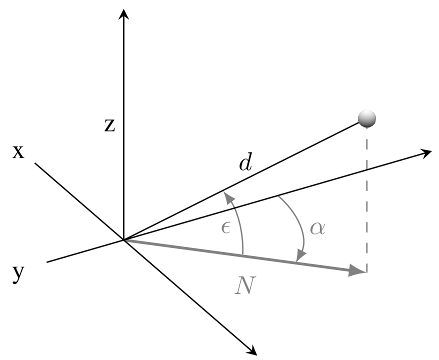

Figure 1. Virtual sound source with spherical coordinates and normal projection N.

Table 1 shows the full list of source attributes and their applicability in the SeamLess system. Only the HOA system is capable of rendering the three-dimensional position of the source. Since the WFS system is working in two dimensions, the elevation, respectively the z component of the position, is ignored and only the x-y components are rendered. The Doppler effect is an inherent part of the WFS rendering algorithm. The pitch shift when moving a source might be undesirable for musical reasons and can be deactivated in the rendering software. WFS is capable of rendering so called focused sound sources. These virtual sound sources are located between the loudspeakers and the listener [4]. The effect of focused sources is a strength of WFS systems, especially for moving audiences, and thus frequently used. Plane wave sources in WFS are characterized through a stable perceived direction of the source independently of the listeners position. For the plane wave, the source the position only influences the phase of the signal. An additional parameter *angle* is set implicitly with the source position so it points to the origin.

Table 1. Accessible properties of rendering systems in the current configuration.

	Position			Doppler	Focused
	X	Y	Z		
WFS	×	×		×	×
HOA	×	×	×		

In the SeamLess system , WFS and HOA can be used in parallel. Sound material can be routed to the different rendering engines with a desired gain through a send bus system, explained in Section 3.1. Although this does not result in a coherent sound field, sounds can be gradually faded between the rendering engines or played on WFS and HOA simultaneously. Besides working with individual sound sources, encoded Ambisonics content can be directly decoded and played over the speakers. This can be artificially created Ambisonics files, as well as recorded content from Ambisonics microphones.

2. HARDWARE

2.1 Loudspeaker Setups

The setups used in the mentioned projects rely on WFS panels by *Four Audio* [1] . Each panel provides eight WFS channels. Each channel is using a column of three tweeters with a horizontal column distance of 10 cm. Four columns share one woofer. The recent version of the panels is equipped with a DANTE interface for receiving the WFS channels from the rendering computers.

2.1.1 TU Studio

The WFS system in the TU Studio has an irregular octagonal geometry with 24 panels, as shown in Figure 2. A 21 channel Ambisonics dome with the geometry shown in Figure 3 uses Neumann KH120A loudspeakers.

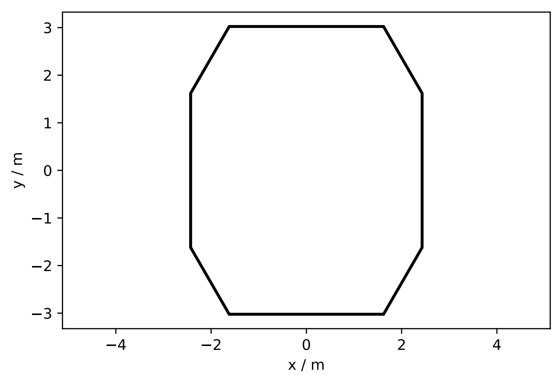

Figure 2. Geometry of the WFS system in the TU Studio.

[1] `http://fouraudio.com/en/products/wfs.html`

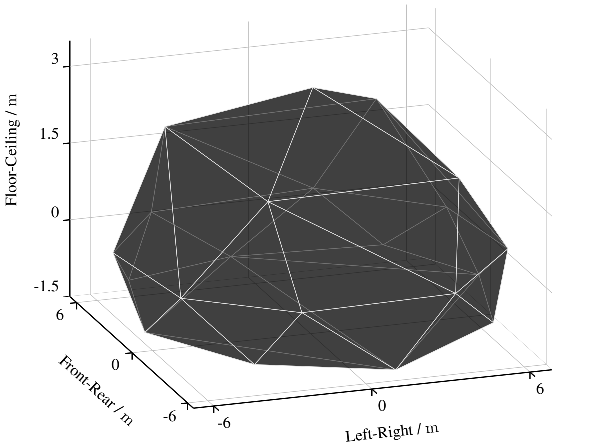

Figure 3. Dome with 21 speakers at TU Studio [5].

2.1.2 *Humboldt Forum*

A dedicated listening room with acoustical treatment was planned for the Humboldt Forum by the *Müller-BBM Holding* [2] and installed by *Neumann & Müller* [3]. Figure 4 shows the top view of the listening room, consisting of two opposing arcs with two entrances. With an area of approximately $7 \times 13\,\mathrm{m}$ it can hold up to 20 standing peoples.

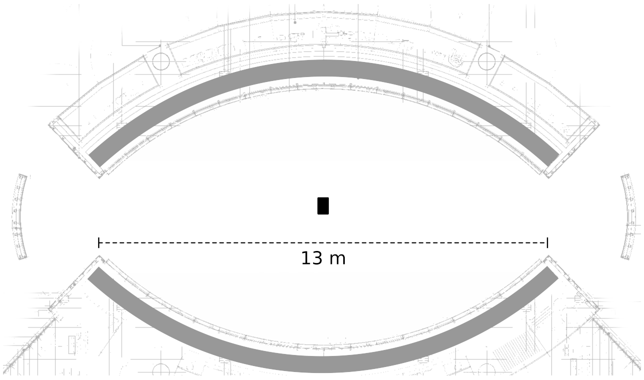

Figure 4. Top view of the listening room at Humboldt Forum with WFS panels (gray) and Ambisonics ceiling speaker (black).

Each arc holds 16 WFS panels, mounted in a continuous ribbon above head height, as shown in Figure 5. This results in a total of 256 WFS channels, respectively 768 tweeters and 64 woofers. 45 *Genelec 8020* speakers, drawn as black rectangles in the top and side view, are used for HOA rendering. They are arranged in three levels ($1.25\,\mathrm{m}$, $1.90\,\mathrm{m}$ $2.5\,\mathrm{m}$) on both arcs. A single ceiling loudspeaker is mounted in the center of the listening area. Additionally, four *Fohhn Arc AS-10* subwoofers are installed at ground level.

2.2 Rendering Servers

The processing of the *SeamLess* system is parallelized on hardware and software level. Work is distributed to several machines, each able to run multiple rendering processes for different loudspeaker sections. In the Humboldt Forum,

[2] https://www.mbbm.de
[3] https://www.neumannmueller.com/en

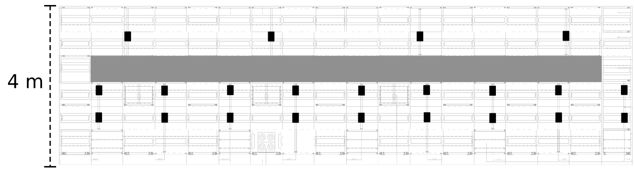

Figure 5. Side view of one arc, showing WFS panels (gray) and Ambisonics speakers (black).

two computers are used for WFS, each rendering 128 WFS channels. Another computer is used for control and HOA rendering.

Dante PCIe cards are used for audio routing between computers and loudspeakers. The models *Digigram LX-Dante* and *Four Audio Dante PCIe* are both suitable for Linux systems. Digigram offers a closed source Linux driver for Ubuntu 20.04 which needs to be updated manually for new kernel versions.

3. SOFTWARE

3.1 System Overview

Professional Linux audio systems offer flexible means for combining different software components, mainly through the *JACK Audio API* [4]. All rendering nodes are thus equipped with Ubuntu (Studio) 20.04. Figure 6 shows the signal flow of a system with a main machine for Ambisonics and routing and two WFS rendering machines, as used in the Humboldt Forum.

A playback computer sends raw audio to all rendering machines and aligned OSC control data for spatialization to the main Linux computer. An OSC routing software converts the incoming generic OSC messages from the playback machine and sends the results to all instances of *SC Mix* and to *cWonder*, the WFS control software.

SC Mix is responsible for Ambisonics encoding and acts as a mixing and distribution component between playback sources and the different rendering units. Send gains to the different rendering units can be controlled for each sound source. Minimal versions of *SC Mix* are thus running on the WFS servers for applying the send gains to the raw audio signals from the playback machine.

This system makes it possible to use different rendering systems simultaneously with a single control interface from the user's perspective. *SC Mix* sets the send gains to the rendering systems according to the OSC Router. The main machine's instance additionally prepares a signal for the subwoofers, encodes an Ambisonics signal from the individual sources and calculates an encoded Ambisonics reverb signal. Separate HOA decoders are used for the mixed signal and the reverb. The *tWonder* WFS rendering instances on the WFS nodes are not directly controlled from the OSC Router but from *cWonder*.

3.2 WFS Rendering

WONDER [6] is a software project for real time wave field synthesis (WFS). It has been designed for running one of the world's largest WFS systems in the lecture hall H 104

[4] https://jackaudio.org/

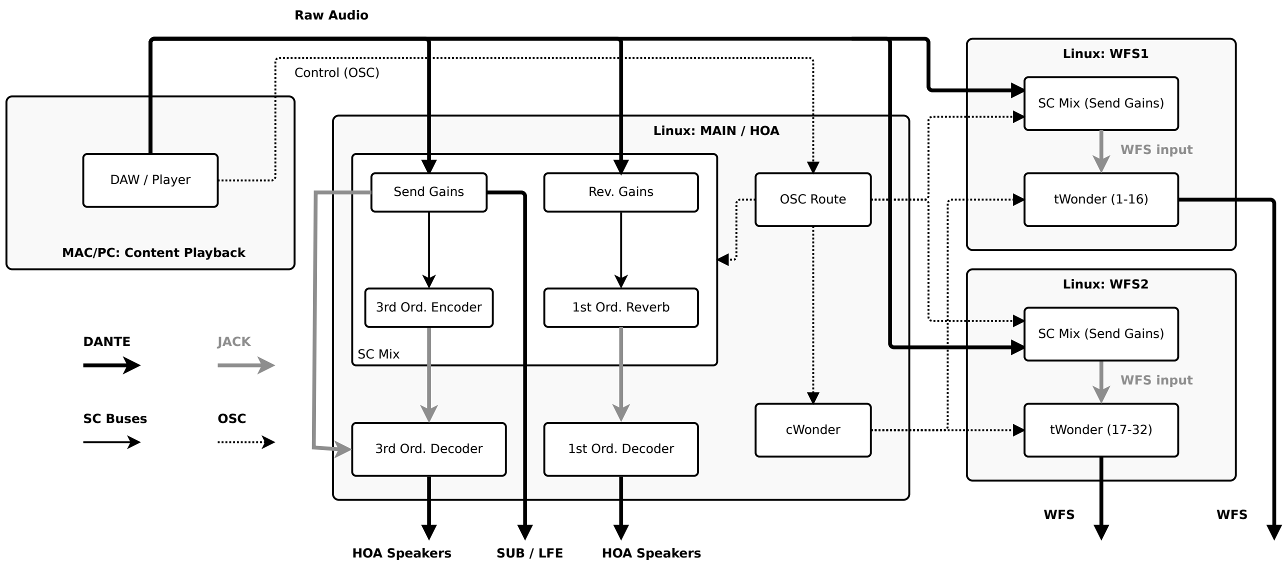

Figure 6. Signal flow with audio and control connections for a setup with two WFS rendering machines.

at TU Berlin, with over 800 channels and an audience capacity of over 600 people. Due to the separation of software modules for control and rendering, WONDER can be used on distributed systems and can thus be scaled to any number of channels. WONDER has also been used as the standard tool in the TU Studio and features components not only for WFS rendering, but also a score player and additional tools.

Equipped with the same WFS panels as the TU Studio, the *I²AudioLab* at HAW Hamburg [5] created *WONDER Lite*, a streamlined version from the original WONDER repository. It includes only the WFS rendering components *cWonder*, *tWonder*, *xWonder* and *libwonder*, minimizing dependencies and easing maintenance as well as further development. Based on this work, additional maintenance and modernizing work was done for this project. Especially the robustness of the startup process was improved. With the introduction of systemd services for the startup of the individual components, the previously used startup scripts become obsolete. As the WFS rendering is controlled through plugins and the OSC Router, the original GUI *xWonder* is not used.

Other WFS rendering solutions tested within this project and used at an earlier stage include the *SoundScape Renderer (SSR)* [7] and the *PanoramixApp* [8]. Each of these alternatives has specific advantages. However, *WONDER* offers the best options for running on a distributed server system as needed.

3.3 Ambisonics Rendering

Ambisonics encoding is realized with the SC-HOA plugins [9] in the *SC Mix* instance on the main rendering machine. For each source, azimuth, elevation and distance can be controlled via OSC commands and all resulting Ambisonics signals are summed inside SuperCollider. This concept has been used in previous projects for distributed spatial audio performances [10].

Decoding of the Ambisonics signals is based on the Ambisonics Decoder Toolbox (ADT) [11], a versatile tool for generating HOA decoders. Making use of Matlab/Octave scripts and the FAUST [12] compiler, it is possible to build decoders for various targets. Initially, the proposed system used decoders in the form of SuperCollider UGens. These have been replaced with standalone JACK clients, which are fed with the encoded Ambisonics signal from the SuperCollider encoder stage.

For *SC Mix* and the HOA encoder, a recent headless version of SuperCollider, as well as the sc3-plugins are built and installed on all machines. As the SuperCollider components are controlled solely via OSC messages and are running in the background, graphic dependencies like X11 and Qt not only represent unnecessary overhead but tend to decrease the audio performance in the absence of a graphic card.

3.4 Reverb

Artificial reverberation can be added from the DAW projects, by sending custom reverb channels to virtual sound sources on the WFS or the Ambisonics system. This was the standard procedure in most projects on the WFS systems in the TU Studio and the auditory. In addition, a parametric Ambisonics reverb is integrated in the proposed software solution. It is based on the Zita [6] reverb implementations in Faust. A SuperCollider UGen is generated with the Faust compiler for creating a first order encoded reverb signal of all sources. This signal is passed to an external first order decoder and subsequently routed to the Ambisonics loudspeakers. Using a parametric reverb allows to change the reverberation characteristics within or

[5] `https://i2audiolab.de/ausruestung/wfs-system/`

[6] `https://kokkinizita.linuxaudio.org/linuxaudio/zita-rev1-doc/quickguide.html`

between projects, without loading additional impulse responses.

3.5 OSC Message Routing

As central interface and connection point between all software modules serves an OSC router and processor written in Python. It is responsible for translating the incoming OSC messages from DAW plugins or other sources to the right format and sending it to the rendering modules. The different software modules it communicates with can be preconfigured or registered as clients during runtime. It differentiates between UI, rendering and data clients. The rendering clients are basically the targets of the OSC-Messages coming in from either UI- or data clients. Data clients send previously recorded or created movement data while UI-clients enable interactive input by the user. Every rendering client is registered with an address, a position format and an update interval. The position of a sound source can be set in any format, such as Cartesian, polar or spherical, and will be converted to the right format when sending it to the client. The maximum send rate for a single source is limited by the update interval and can also be set individually for every rendering client. Hence the WONDER software, which has its own position interpolation methods, can be fed with a lower position data rate than the Ambisonics renderer which works better with a high position data rate input. While UI and rendering clients are technical treated the same and are informed about every change in the source data, the data clients are only getting changes coming from UI clients to avoid a loop of OSC messages. For easier handling, inputs of UI clients can block the input of data clients for a short amount of time, since some OSC automation plugins are constantly sending their state.

3.6 Production and Playback

Since the rendering system is controlled in real time with OSC messages, it is not bound to specific software for production and playback, as long as audio and related control data are streamed synchronously. However, for reasons of accessibility, a DAW based approach with plugins for OSC automation is proposed. Reaper was chosen as the main DAW for content production and playback. One reason for this is the high flexibility in channel routing when working with multi-channel content. Audio files with up to 64 channels can be created and processed in Reaper, allowing tracks with up to 64 channels and the same number of outputs. This makes it also ideal for Ambisonics content that may have a high channel count depending on its order. Further, Reaper allows embedding plugins and automation trajectories into single audio items inside a project. The final distribution format is a rendered multichannel audio file, including the embedded automation data, which can be used for the content playback and arrangement. By using the Reaper scripts, the rendering of audio output and the export of automation data can be fully automated.

The approach of separating rendering software from the DAW relies on plugins capable of sending OSC commands, based on automation trajectories. Each audio track in a project is equipped with one instance of such plugin, allowing to automate its source position and other attributes. A working solution is provided by IRCAM, namely *OSCar* [7], the successor of the *ToscA* plugin. *OSCar* has been designed for controlling the *PanoramixApp* from a DAW. Alternatively, the free software plugin *osccontrol-light* [8] has been tested. Albeit currently less feature-rich, the plugin code is open source and can be adjusted to meet the requirements. Both solutions are general purpose tools and can be configured to send and receive the relevant OSC messages through configuration files. Finally, a dedicated plugin has been developed for this project and is included in the software repository. It is application-specific and needs no additional configuration, besides the target IP address and port.

3.7 Subsystem Configuration and Integration

The startup of WONDER was originally managed with shell scripts. Those scripts relied on a specific order of execution. If the startup failed at some point, all systems had to be restarted. The scripts were responsible for the startup of the entire system, including JACK. For this they used additional configuration files. The configuration is now stored in the system-wide directory `/etc/wonder`. This also includes the speaker locations for the WFS renderer as well as the remaining startup script configuration. All other configuration files, including JACK configuration and connections, are located in `/etc/seamless`.

systemd services offer a more robust startup process in comparison to a cascade of shell scripts. All services can be started and stopped independently. They can define requirements, such as network and audio interfaces, and can be configured to start on system boot. There are system services now for JACK, *tWonder*, *cWonder*, the Ambisonics decoders, *SC Mix*, the OSC Router and *aj-snapshot*. Running JACK as a system service resulted in a conflict with Ubuntu Studio's autojack when logging in. To prevent this, the ubuntustudio-control package needs to be removed.

Improving the startup process significantly increases the robustness and reliability of WONDER. A failed *tWonder* instance is now able to reconnect to *cWonder*. The aj-snapshot service is used to manage all JACK connections. JACK connections can be stored and reloaded as snapshots for different use cases. The service runs as a daemon and connects new jack clients, if applicable.

4. WORKFLOW AND APPLICATION

4.1 Combining WFS and HOA

When using Ambisonics and WFS in a combined system, their individual strengths can be used for specific effects. Since Ambisonics comes with inherent means of recording, it is well suited for capturing actual soundscapes on site with dedicated Ambisonics microphones. In the case of the listening room at Humboldt Forum this seems to be a recurring concept, since artists often provide field recordings from specific sites.

WFS, on the other hand, is well suited for creating highly locatable sound events from isolated recordings, as achie-

[7] `https://forum.ircam.fr/projects/detail/oscar/`

[8] `https://github.com/drlight-code/osccontrol-light`

ved by close up miking. Especially for a moving audience, as in the listening room, this draws the attention to such focused sound sources and creates the illusion of virtual objects in the listening space. An advantage of WFS comes from the fact that human hearing capabilities have the highest resolution on the horizontal plane while sources coming from above and below the ear-level can not be localised with such a high precision. Having the WFS-system with its very high spatial resolution on the horizontal plane, this relationship can be used. By connecting the elevation property of a source directly to the gain send to the different systems, the combined use of both systems can be simplified from the user's perspective. The send-based approach allows the continuous fading of sound sources between the different rendering systems. Since they share the same spatial parameters, the position of all virtual sound sources are identical, apart from differences between 2D and 3D rendering.

4.2 User Control

In previous productions with spatialization systems it has been found that the demands on user interfaces strongly vary among composers and other users. Although the included plugin comes with a simple graphical spatialization interface, custom solutions are often desired. Since the OSC router accepts various different message formats, it is open for custom interface and control approaches and also provides appropriate feedback for graphical user interfaces. Individual solutions for the different projects can thus be implemented easily and might lead to a repertoire of different control possibilities. Nevertheless, the design of an intuitive user interface, fitting most use cases, is a continuing aspect of the project.

5. CONCLUSIONS

The proposed software system allows the seamless integration of different spatial rendering approaches and multiple loudspeaker systems. Although the long term operation is yet to be evaluated, the first application in a production for the Humboldt Forum delivered a proof of concept, allowing a composer to realize his ideas on the spatial arrangement and evolution with the help of the engineers.

In its recent state, the system is still under development and specific aspects are subject to improvements. Due to the modularity of the system, alternative rendering software can always be tested, compared and included. For instance, the *Ambisonics Toolkit (ATK)* for SuperCollider [9], which is part of the SC3-Plugins, was recently upgraded to also allow Higher Order Ambisonics. Other possible improvements include parametric reverb with directional early reflections for an increased plausibility or the use of high quality Ambisonics room impulse responses.

Based on free and open source software, the proposed system is an open source project itself and can be accessed through the related GitHub repository. [10] A detailed documentation with instructions for musicians, as well as setup instructions for system administrators can be found in the corresponding GitHub pages. [11] Due to the flexibility, this solution might thus be applicable in related setups and use cases.

6. REFERENCES

[1] A. Lindau, R. Kopal, A. Wiedmann, and S. Weinzierl, "Räumliche Schallfeldsynthese für eine musikethnologische Ausstellung: Erfahrungen aus Produktion und Rezeption," in *Proceedings of the Inter-Noise 2016: 45th International Congress and Exposition on Noise Control Engineering: towards a quiter future*, 2016.

[2] M. Kronlachner, "Plug-in suite for mastering the production and playback in surround sound and ambisonics," *Gold-Awarded Contribution to AES Student Design Competition*, 2014.

[3] K. L. Hagan, "Textural composition: Aesthetics, techniques, and spatialization for high-density loudspeaker arrays," *Computer Music Journal*, vol. 41, no. 1, pp. 34–45, 2017.

[4] H. Wierstorf, A. Raake, M. Geier, and S. Spors, "Perception of focused sources in wave field synthesis," *Journal of the Audio Engineering Society*, vol. 61, no. 1/2, pp. 5–16, 2013.

[5] H. von Coler, "A System for Expressive Spectro-spatial Sound Synthesis," Ph.D. dissertation, 2021.

[6] M. A. Baalman and D. Plewe, "WONDER - a Software Interface for the Application of Wave Field Synthesis in Electronic Music and Interactive Sound Installations," in *Proceedings of the International Computer Music Conference*, 2004.

[7] J. Ahrens, M. Geier, and S. Spors, "The SoundScape Renderer: A unified spatial audio reproduction framework for arbitrary rendering methods," in *Proceedings of the 124th Audio Engineering Society Convention*. Audio Engineering Society, 2008.

[8] T. Carpentier, "Panoramix: 3d Mixing and Postproduction Workstation," in *Proceedings of the International Computer Music Conference (ICMC)*, 2016.

[9] F. Grond and P. Lecomte, "Higher Order Ambisonics for SuperCollider," in *Proceedings of the Linux Audio Conference 2017*, 2017.

[10] H. von Coler, N. Tonnätt, V. Kather, and C. Chafe, "SPRAWL: A Network System for Enhanced Interaction in Musical Ensembles," in *Proceedings of the Linux Audio Conference*, 2020.

[11] A. Heller and E. Benjamin, "Design and implementation of filters for Ambisonic decoders," in *Proceedings of the 1st International Faust Conference (IFC)*, 2018.

[12] Orlarey, Yann and Fober, Dominique and Letz, Stephane, "Syntactical and semantical aspects of Faust," *Soft Computing*, vol. 8, no. 9, pp. 623–632, 2004.

[9] `https://www.ambisonictoolkit.net/documentation/supercollider/`

[10] `https://github.com/anwaldt/seamless/`

[11] `https://anwaldt.github.io/seamless/`