INTERACTIVE-ALGORITHMIC CONTROL OF SOUND SPATIALIZATION

Name of author

Address - Line 1
Address - Line 2
Address - Line 3

ABSTRACT

We present recent works carried out in the OpenMusic computer-aided composition environment for connecting compositional processes with spatial audio rendering. In particular, we developed tools to explore new modalities for manipulating sound spatialization data with a combination of algorithmic processes and interactive control using remote interfaces.

1. INTRODUCTION: AUTHORING TOOLS FOR SPATIAL AUDIO

Authoring tools for spatial audio are naturally influenced by the orientation of their host environments or of the technological frameworks they fit in, be it from computational, representational or user interaction points of view. For instance, the Spat library [1] provides rich and multi-fold interfaces for monitoring real-time spatialization in Max, whereas spatialization tools in OpenMusic [3] integrate offline spatial audio control and rendering in computer-aided composition (CAC) processes and emphasize the expressivity and computational power of algorithmic specifications [4]. Other examples include for instance SSMN (Spatialization Symbolic Music Notation [5]), which embeds spatial cues and trajectories in a score editor and connects it to a spatial rendering system, or ToscA [6], a new plugin providing a straightforward connection of DAWs to the Spat real-time DSP and interfaces. Other approaches focus on gestural controllers to manipulate complex sets of parameters with rich and expressive possibilities [7, 8] but they are generally limited to performance application and composers can rarely take advantage of such devices or interactions.

Despite this diversity of orientations a real gap exists in practice between the authoring and spatialization tools used during early compositional stages, and those actually used in production, either on stage or during recording sessions. This gap is a challenge for computer music research: it is hard to compose music using spatial structures and movements without a structured representation of time (as in scores or in CAC environments), and equally hard to think and design spatialization without efficient audio-visual feedback.

A common solution is to use a multi-layered systems separating spatial rendering engines from the authoring tools that produce control data [9, 5]. In such multi-layered systems, modularity is paramount for benefiting from the different representations and computation paradigms (for instance, combining explicit representations of time and real-time rendering). It implies seamless protocols for data exchange, which can be carried out either via file I/O or via networking (using OSC [10] or other dedicated protocols [11]).

Similar concerns have driven recent developments of the OpenMusic computer-aided composition environment [12]. File storage based on the SDIF format [13] and OSC-based communication systems have been proposed to connect trajectories and other spatial specification data produced in OpenMusic to spatial rendering tools, either in an off-line mode (via SDIF and audio files produced and collected in the CAC environment) or in a real-time flavor (via dedicated players streaming the generated timed information to Max/*spat*$\sim$ for rendering). We present here some related tools and applications developed as part of our recent research.

2. RESEARCH CONTEXT: INTERACTIVE COMPUTER-AIDED COMPOSITION SYSTEMS

It is common to differentiate computer-aided composition from interactive data/audio environments considering their computational models (generally, offline vs. real-time). Some recent projects however bring up a more ambiguous vision and a flexible positioning of computer music environments with regards to the notions of musical structure and interaction [14, 15].

"Reactive processes" in OpenMusic [15] combine the formal approaches of computer-aided composition systems, based on off-line, functional-style programming, with reactive approaches inspired by interactive musical systems. This project highlights new perspectives linking off-line computations to the unfolded time of musical executions, allowing changes or events occurring in visual programs (or in the data they contain) to produce series of updates and evaluations. These events can come from user actions or external sources (e.g. MIDI or UDP/OSC ports), so that bilateral communications can be established between compositional programs and remote applications or devices.

Reactive CAC processes participate in a structured interaction with their context, which can take place either in a compositional perspective (in the processes leading to the generation of musical material) or in a context of performance. The CAC environment therefore becomes a part of a larger-scale system composed of several interfaces, tools and rendering engines, and can be driven by events or interactions produced in this system. For instance, an OpenMusic visual program can collect data from a music performance, compute representations on the fly or process this data to generate new musical material (as in an automatic accompaniment or improvisation system where sequences played by a musician are analysed and recombined to produce other parts [16]). More specific examples related to the control of sound spatialization will be given further on.

3. TOOLS AND TECHNOLOGICAL FRAMEWORK

Currently the main tools available for the control sound spatialization in OpenMusic are:

OM-Spat. This library provides a set of tools centred around the *spat-matrix* object: an array of parameters specifying the position, orientation, aperture, reverberation of an arbitrary number of sound sources. All the parameters can be either static or time-varying data. The matrix is converted to linear streams of control frames written in an SDIF file. The spat command line tool shipped with the library is responsible for the (non real-time) spatialization rendering: it processes input audio files according to the SDIF control file and produces a multi-channel bounce.

OMPrisma. This library is an extension of OMChroma, a framework for creating sounds with Csound, using other matrix structures controlling the different parameters of the different instances of Csound synthesis patches. With OMPrisma it is possible to connect spatial processing to these patches, and therefore to compute and set spatial parameters for every component of a synthesis process (e.g. every partial, sound grain, etc.) – an approach referred to as "spatial sound synthesis" [2].

An important aspect in the definition and programming of complex spatial scenes and trajectories in compositional processes is the management of time in relation to the spatial data (and possibly to other external parameters). This is also where computer-aided composition systems can make the difference as compared to other authoring tools anchored in real-time DSP frameworks. Curves and trajectories are represented as standard objects in the OpenMusic visual programming environment and can be used for instance to parametrize the control matrices in OM-Spat/OMPrisma. Specific editors have been developed to represent the data in 3D and help editing it using projection planes. These object can be set and edited manually, or programmed and generated by arbitrary complex algorithms related to the compositional processes (see Figure 1). 3D points can also carry explicit time information. If not, their time-stamp is interpolated from surrounding points' timing at rendering (e.g. when the curves are converted to sequences of SDIF frames).

We have previously noticed that communication with external devices and applications was a means to bridge the gap between formal/algorithmic compositional tools and interactive/real time spatial audio renderers. Such communication can take place at various stages of the composition and processing of spatial parameters: during the authoring phase, where trajectories and other parameters are defined, or during the rendering when this data is transferred to DSP systems and integrated with real-time interactions. Accordingly, several external tools are currently available and connected to the computer-aided composition framework, such as:

Spat-SDIF-Player. This application is a Max standalone communicating with OM to load the generated SDIF files. It provides simple cues to stream the contents (mostly, the position of the sound sources) as OSC messages encoded according to the SpatDIF specification. The OM-generated data can then be rendered by the Spat~ or any system able to receive an interpret these messages.

Trajectoires. This web application runs either on desktop computers, smart-phones or tablets [17]. It allows to draw and edit trajectories with finger-based interaction and communicates bi-directionally with OM (or other applications) through the OSC protocol. It also acts as a mobile remote control to stream the data in real-time (see Figure 2).

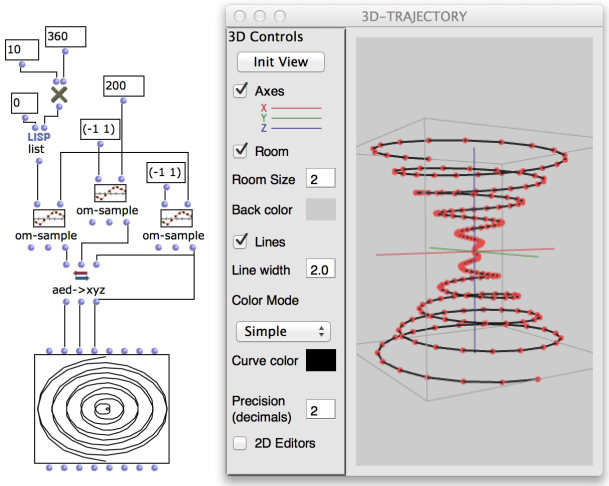


Figure 1: Generation of 3D trajectories in OpenMusic.

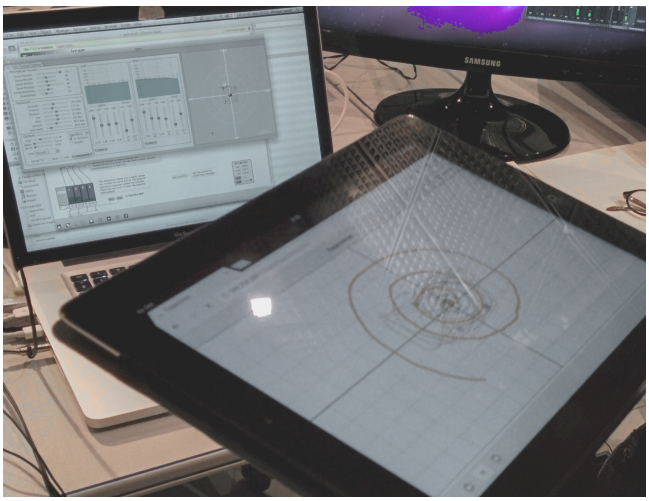


Figure 2: Using the *Trajectoires* mobile application to control spatialization in Max/*Spat*.

4. APPLICATIONS: IN-LINE ALGORITHMIC PROCESSING OF TRAJECTORIES

The following examples show potential applications involving the proposed computer-aided composition tools (and in particular the *Trajectoires* mobile application) in reactive visual programs, and how interesting workflows can emerge at the crossways between formal composition and interaction. We will consider a work session where a composer uses OpenMusic (OM) for composing spatial structures (we can imagine that other aspects of his/her compositional process are also carried out in this environment and related to these spatial structures), *spat~* within Max for rendering and monitoring the spatialization, and the *Trajectoires* application loaded on a mobile device (iPad or equivalent).

It is straightforward to design arbitrary processes generating trajectories in OM (see for instance Figure 1) and to send them to the *Trajectoires* application in order to constitute a dictionary of spatial data, which can then be selected, composed and manually transformed to constitute some of the spatial scene components.

Conversely, the trajectories produced in the mobile application can be sent over the network and received both in the spatial rendering environment and in OM. A reactive OM patch as exemplified on Figure 3 receives and filters incoming data, and processes it through transformation algorithms (in this case, a simple rotation, but we can imagine any transformation related to a compositional process). As the patch is reactive, this transformation operates immediately and automatically as soon as a trajectory is received, and the internal data containers and editors are updated accordingly. Sending back the data to the mobile application can also be part of the reactive process (as it is the case in Figure 3) and the mobile application will immediately receive and store a new transformed trajectory.

Similar processes can be applied iteratively to generate sets of trajectories (e.g. in this case rotated with different angles) from a single input from the drawing application. The generated curves sent back to the mobile application can be assigned to different sources and "played", that is, streamed together as OSC messages to control a real-time spatial audio renderer.

Figure 4 shows an example involving slightly more complex interactions: the *route-osc* box in the OM patch routes the downstream reactive notifications and updates depending on an identifier assigned to the received trajectory. A first trajectory ("source1/traj") is just stored, while upon reception of a second trajectory ("source10/traj"), an interpolation process is executed to compute a number of intermediate trajectories. Figure 4(b) shows the interpolated curves received in the *Trajectoires* interface.

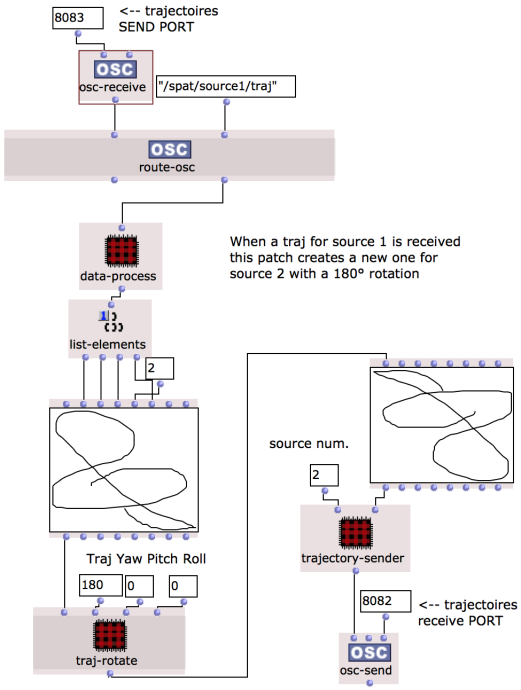


Figure 3: Reactive processing of trajectories in OpenMusic. The dark-framed boxes are *reactive* to upstream changes. The *trajectory-sender* sub-patch formats OSC messages to be sent via UDP by the *osc-send* box.

5. CONCLUSION

The examples presented in this paper illustrate the potential of computer-aided composition programs considered as reactive components in the general temporality of a compositional process. The OpenMusic environment proposes a powerful and visual programming framework, which could also be involved during performances or in other interactive situations (e.g. in sound installations) in order to process, transform, expand, reinterpret dynamically any incoming data to generate arbitrarily complex controllers and data to the real-time DSP environments.

6. REFERENCES

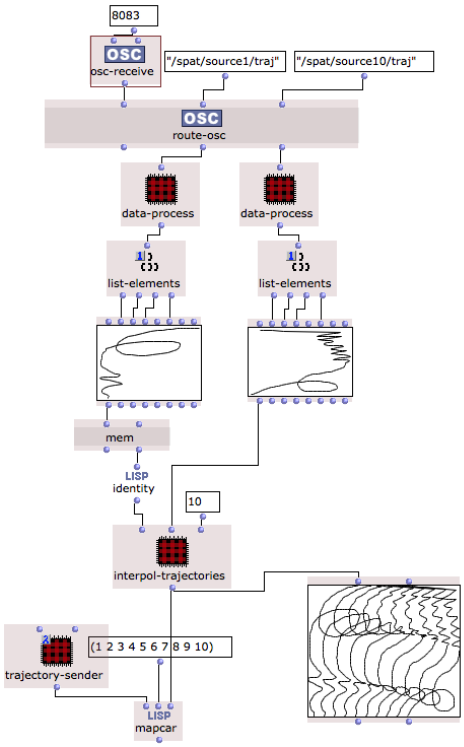
[1] T. Carpentier, M. Noisternig, and O. Warusfel, "Twenty years of Ircam Spat: looking back, looking forward," in *Proceedings of the International Computer Music Conference*, Denton, United States, 2015.

[2] M. Schumacher and J. Bresson, "Spatial Sound Synthesis in Computer-Aided Composition," *Organised Sound*, vol. 15, no. 3, 2010.

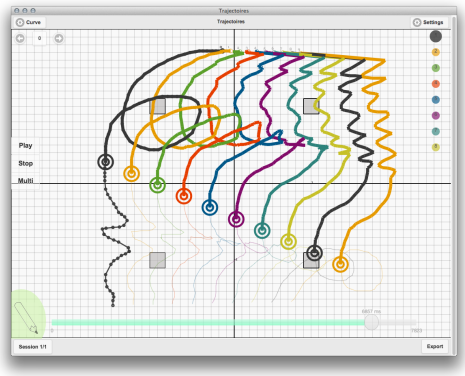
[3] J. Bresson, C. Agon, and G. Assayag, "OpenMusic. Visual Programming Environment for Music Composition, Analysis and Research," in *ACM MultiMedia (MM'11) OpenSource Software Competition*, Scottsdale, United States, 2011.

[4] J. Bresson, "Spatial Structures Programming for Music," in *Proceedings of the Spatial Computing Workshop (SCW) – Co-located w. Autonomous Agents and MultiAgent Systems (AAMAS)*, Valencia, Spain, 2012.

[5] E. Ellberger, G. Toro-Perez, J. Schuett, L. Cavaliero, and G. Zoia, "A Paradigm for Scoring Spatialization Notation," in *Proceedings of the International Conference on Technologies for Music Notation and Representation - TENOR*, Paris, France, 2015.



(a) Interpolation between trajectories in OM. The first received trajectory is memorized (*mem*) and the interpolation is triggered upon reception of the second trajectory.



(b) Reception and monitoring in *Trajectoires*.

Figure 4: Interactive computation of interpolated trajectories.

[6] T. Carpentier, "ToscA: an OSC communication plugin for object-oriented spatialization authoring," in *Proceedings of the International Computer Music Conference*, Denton, United States, 2015.

[7] J. J. Nixdorf and D. Gerhard, "Real-time sound source spatialization as used in challenging bodies: implementation and performance," in *Proceedings of the 2006 conference on New interfaces for musical expression*. IRCAMCentre Pompidou, 2006, pp. 318–321.

[8] M. T. Marshall, J. Malloch, and M. M. Wanderley, "Gesture control of sound spatialization for live musical performance," in *Gesture-Based Human-Computer Interaction and Simulation*. Springer, 2009, pp. 227–238.

[9] C. Bascou, "Adaptive Spatialization and Scripting Capabilities in the Spatial Trajectory Editor Holo-Edit," in *Proceedings of the Sound and Music Computing Conference*, Barcelona, Spain, 2010.

[10] M. Wright, "Open Sound Control: An Enabling Technology for Musical Networking," *Organised Sound*, vol. 10, no. 3, 2005.

[11] N. Peters, T. Lossius, J. Schacher, P. Baltazar, C. Bascou, and T. Place, "A Stratified Approach for Sound Spatialization," in *Proceedings of the Sound and Music Computing Conference*, Porto, Portugal, 2009.

[12] J. Bresson and M. Schumacher, "Representation and Interchange of Sound Spatialization Data for Compositional Applications," in *Proceedings of the International Computer Music Conference*, Huddersfield, United Kingdom, 2011.

[13] D. Schwartz and M. Wright, "Extensions and Applications of the SDIF Sound Description Interchange Format," in *Proceedings of the International Computer Music Conference*, Berlin, Germany, 2000.

[14] A. Agostini and D. Ghisi, "Real-time computer-aided composition with bach," *Contemporary Music Review*, vol. 32, no. 1, 2013.

[15] J. Bresson, "Reactive Visual Programs for Computer-Aided Music Composition," in *IEEE Symposium on Visual Languages and Human-Centric Computing*, Melbourne, Australia, 2014.

[16] J. Nika, D. Bouche, J. Bresson, M. Chemillier, and G. Assayag, "Guided Improvisation as Dynamic Calls to an Offline Proccess," in *Proceedinsg of the Sound and Music Computing conference*, Maynooth, Ireland, 2015.

[17] X. Favory, J. Garcia, and J. Bresson, "Trajectoires : une application mobile pour le contrôle et l'écriture de la spatialisation sonore," in *Conférence Francophone sur l'Interaction Homme-Machine*, Toulouse, France, 2015.