REPRESENTATION AND INTERCHANGE OF SOUND SPATIALIZATION DATA FOR COMPOSITIONAL APPLICATIONS

Jean Bresson

STMS: IRCAM-CNRS-UPMC
1, place I. Stravisnky 75004 Paris, France

Marlon Schumacher

IDMIL – DCS – CIRMMT, McGill University
555 Sherbrooke St West, Montreal, QC, Canada

ABSTRACT

We present recent works on the representation of spatialization data for the authoring, interchange and rendering of spatial audio in musical contexts. These works are mainly implemented in the OpenMusic computer-aided composition environment. High-level structures created in compositional processes are represented by matrices, stored using a standardized interchange format and finally read and interpreted by external tools for offline and real-time rendering.

1. INTRODUCTION

With the growing number and accessibility of sound spatialization technologies, space is now a critical dimension in compositional systems. The representation of spatial data in compositional applications must fulfill at least two complementary constraints: On the one hand, it must be abstract enough to be handled in musical contexts, close to cognitive and symbolic models; On the other hand, it must define potentially large amounts of high-resolution control data, as required for spatial sound rendering.

At the same time, the need for generic, standardized representations and interchange solutions, independent from particular authoring or rendering environments, arises from the growing diversity of the musical practices and applications involving spatial composition.

In this paper we present recent works on the representation and rendering of spatial description data in musical environments. We propose a solution for modelling and interchange of spatialization data between musical systems based on the SDIF format, and show concrete applications and new developments in the OpenMusic and Max/MSP environments.

2. REPRESENTATION OF SPATIAL SCENES

Let us consider an arbitrary number of sound sources and their respective set of spatial parameters (such as position, orientation, directivity, etc.) abstracted from the reproduction conditions (e.g. from a given loudspeaker setup). Previous works on control of sound spatialization in the computer-aided composition environment OpenMusic [2] have employed matrix structures for the descriptions of such spatial scenes [11, 15, 6]. Associated to high-level functional specifications, these structures allow to unite symbolic musical representations and concrete sound rendering data.

Figure 1 shows a patch using the new OpenMusic *OM-Spat* library. A spatialization matrix (a) represents the different components of the scene (sound sources) associated to a set of chosen parameters describing the spatialization of these components. OpenMusic matrices provide powerful symbolic features and control accuracy for authoring and composition [1]. It is possible, for instance, to assign a common trajectory to the different components of a spatial scene, to set relations between trajectories, or to define algorithmic processes for the dynamic generation of these trajectories.

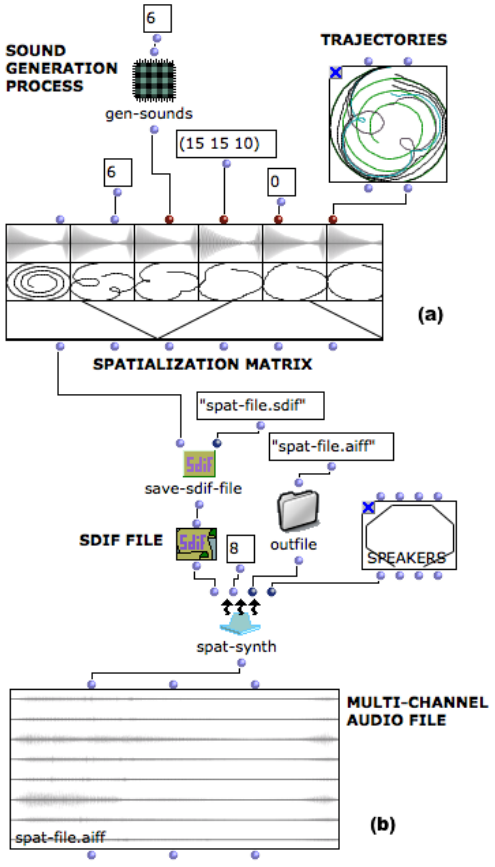

Figure 1. Spatial scene representation and synthesis in OpenMusic (*OM-Spat*). The scene description matrix (a) containing 6 sources and spatial parameters is rendered to a multi-channel sound file (b).

3. FROM SYMBOLIC TO SIGNAL REPRESENTATIONS

Compositional environments are historically concerned with the representation and manipulation of symbolic materials rather than the concrete control signals in their final resolution—as required by digital controllers and DSP systems. Indeed, one of the main objectives in these environments is to provide interfaces for intuitive/musical setting and editing of high-level musical representations, as well as possibilities for "functional" (or algorithmic) processing (e.g. applying transformations, time-compression/stretching, interpolations, etc.) Yet in this context, a particular concern has to be given to the temporal aspects of the spatial parameters since the symbolic, high-level specifications are eventually "compiled" into accurate control signals for the control of spatial sound rendering processes.

For the purpose of this discussion we will take the example of spatial trajectories, defined as sequences of three-dimensional coordinates controlling the evolution of sound source positions. Same principles could however apply to static positions and to other types of descriptors such as source orientation, directivity or room parameters.

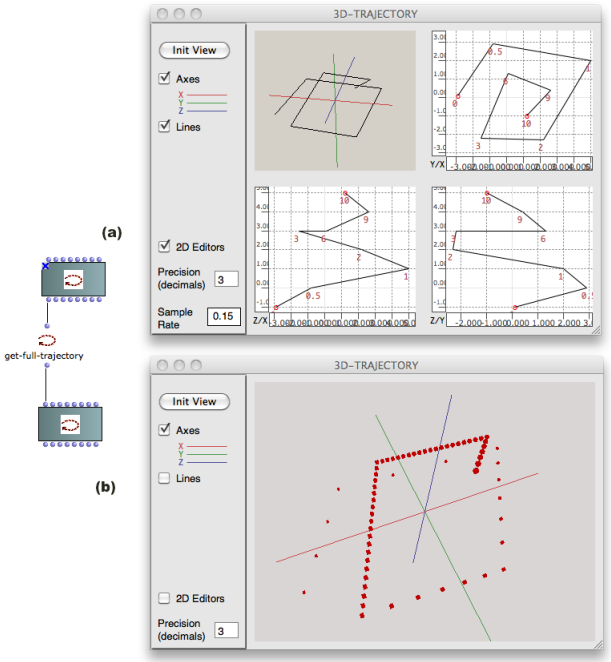

Figure 2. 3D-trajectories in OpenMusic. (a) High-level specification (a 10-points time-tagged curve) is converted to a higher-rate sampled trajectory (b).

The *3D-trajectory* is a new object we created in OpenMusic, combining high-level/symbolic representation and higher-resolution specification. The *3D-trajectory* is initially defined by a set of time-tagged points in 3D euclidean space. 3D points may *or not* be assigned temporal information; all missing time-tags will be deduced by interpolation when rendering the trajectory. The trajectories can be edited thanks to 3D editors allowing to draw or modify the positions or time-tags of the points. The tools for the generation and processing of curves available in the compositional environment allow to create or transform them using algorithmic means, or starting from reduced sets of points used as initial data for sampling and interpolation processes (at different spatial or temporal resolutions).

A *sample-rate* parameter in the *3D-trajectory* object allows to produce, if required, a full-precision trajectory by sampling the original one at a given rate. The sample rate itself can be a varying parameter and change over the different segments of the original trajectory: Figure 2 shows two *3D-trajectory* objects and respective editors, and the conversion of a "control trajectory" to a full-sampled trajectory suitable for the control of a sound spatialization system. Depending on the degree of abstraction required at defining a spatial sound scene, both types of trajectory could be used and assigned to a sound source in a scene representation such as the one in Figure 1.

4. A STORAGE SOLUTION USING SDIF

The authoring, rendering and reproduction of spatial sound scenes often take place in distinct environments and can involve varying hardware setups. The multiplicity of existing concepts and technologies in the field of spatial audio therefore elicits the need for standardized descriptions and interchange formats (see the dedicated panel during the 2008 International Computer Music Conference [9]). A number of specifications have so far been proposed for the description of spatial scenes, based for instance on the MPEG or XML standards. Proposed storage formats (e.g. ASDF [7], also more general formats like VRML/X3D) are mostly based on high-level specifications about spatial scenes setup, hierarchical structures and transformations.

Our interest, at the contrary, is to describe spatial sound scenes from a "denotative" point of view, that is, providing descriptions of time-sampled control data in their final resolution. Indeed an important idea in our approach to sound synthesis and spatialization is that compositional environments should not only consider high-level parameters but also encompass as far as possible the final sound rendering aspects [5]. From this point of view, as shown above, the sampling parameters can be integrated in the control environment as "compositional" variables, and the scene descriptions would rather consist of final time-sampled control data than abstract specification statements. Consequently, these descriptions can be directly rendered without further interpretation or processing by spatialization systems.

We decided to use the Sound Description Interchange Format (SDIF [16]) as a container for the encoding and interchange of spatialization data. SDIF is used to represent and store sound description data in OpenMusic [4], as well as in a growing number of other music and sound processing environments, libraries and bindings. This format is intended for efficient, low-level description. It is not bound to a fixed sample rate, which makes it well-suited for applications in which time resolution is a vari-

able parameter (and not determined by the format). As we will show hereafter, it provides the flexibility and structure required for multiple source descriptions with multiple descriptors and heterogeneous temporal scales or resolutions.

SDIF is based on matrices (hence an a priori suitable structure for our conception of spatial scenes) describing the values of a number of parameters (called "fields") for a number of components. The matrices are assigned a "type" (a signature of 4 ASCII characters) and are included in time-tagged frames. Frames also have a type signature and a stream ID, which allows to describe and discriminate interleaved frame streams and to encode multiple simultaneous data flows.

A number of standardized SDIF types exist, which correspond to common sound descriptions used in the computer music community (spectral descriptions, sinusoidal tracks, frequencies, energy, etc.). SDIF provides a straightforward protocol to add new type definitions or extensions, which can be declared in file headers so that reader applications can dynamically register, identify and interpret (or not) the data contained in the SDIF frames and matrices.

4.1. SDIF Spatial Descriptions

We defined a number of basic SDIF type conventions for storing sound source trajectories and spatial sound scenes in general. Frames of type "XSRC"[1] represent the spatial parameters of sound sources. They contain matrices of type "XCAR" (for *Cartesian* coordinates) describing the position of these sources.[2] Figure 3 shows a sequence of SDIF frames describing the evolution of a sound source's position (i.e. its trajectory).

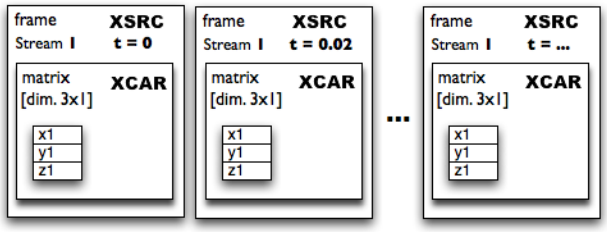

Figure 3. Sequence of SDIF frames describing the trajectory of a sound source.

Note that time-tags (t) are specified for every frame: SDIF does not have a notion of sample rate and the frames temporal localization can be freely assigned at any required resolution.

Each SDIF frame may contain one or several matrices: The source parameters can be extended with other descriptors, such as orientation or aperture, declared in specific SDIF matrices. Figure 4 shows the example stream from Figure 3 with additional orientation matrices (type "XORI") in some of the SDIF frames. Note that a given type of frame does not necessarily need to contain all possible matrices at a time, which allows to set different temporal granularities for each spatial descriptor.

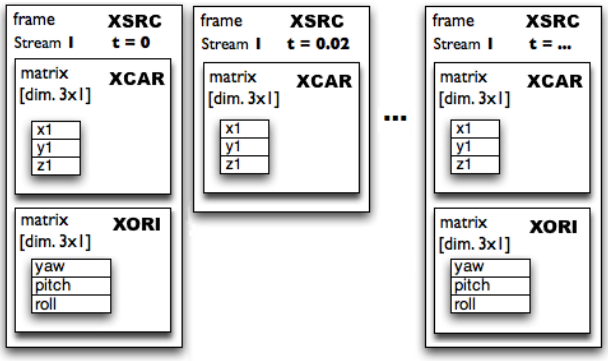

Figure 4. Sequence of SDIF frames describing the trajectory and orientation of a sound source.

Room effects can be described in matrices as well, included either in the source description frames (if related to a particular source), or in distinct frame streams (to specify global ambient characteristics and their evolution).

4.2. Multiple Sources and Scene Graph Modelling

Spatial scenes including multiple moving sound sources can be structured in different ways using this format.

Considering a number of self-consistent sound sources (typically, audio files, as in the case of the spatialization matrix in Figure 1), we might be interested in generating separate control streams for the spatial attributes of the different sources, since each of them might evolve at a different rate or temporality: Some sources may be still (and described with a single or with a few number of frames) while other sources would move in more complex ways and be described at higher resolution. Figure 5 illustrates this case of multiple sources, where each source is controlled and encoded in a separate frame stream.

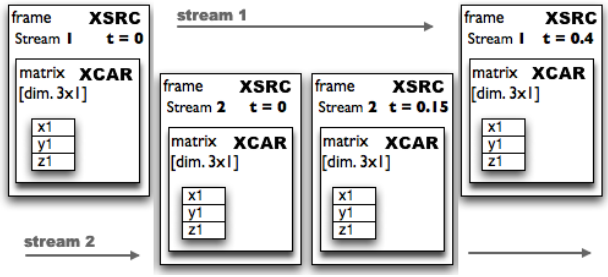

Figure 5. Sequence of SDIF frames describing the positions of two sources moving independently (streams 1 and 2).

[1] According to the SDIF conventions, standardized type signatures start with a "version" number ("1", most of the time) while more specific and experimental type signatures start with "X".

[2] The type signatures here are given as examples; Different naming conventions may be adopted in the future. Also note that other compatible conventions could be adopted for other coordinates system, e.g. XCYL or XSPH for cylindrical or spherical.

When the sources are conceptually linked together or form a compound sound source, as in the case of the primitive components of a sound synthesis process (partials, grains, etc.—which we refer to as *spatial sound synthesis*, see [15]) it is probably more convenient (and efficient) to consider a single frame stream in which matrices of dimensions $3 \times n$ would contain the full set of scene components' descriptions (3 Cartesian coordinates are specified for n sources). These descriptions are then packed in a single data unit and perfectly synchronized. Figure 6 illustrates this second case.[3]

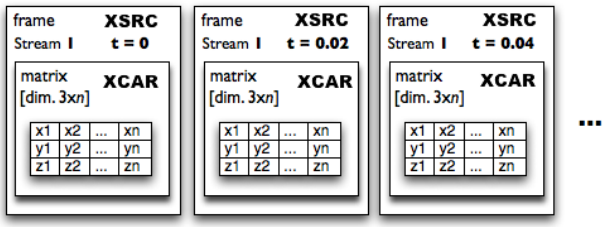

Figure 6. Sequence of SDIF frames describing the positions of n sound sources. In each frame, the n positions are described in a same matrix.

Note that the two cases envisaged can also be combined, which would allow to group the elements of a scene into different data streams independent from one another.

5. FROM COMPOSITIONAL DESCRIPTIONS TO SPATIAL RENDERING

Spatial sounds and spatial sound scenes in general can therefore be described in matrix structures, temporally organized following variable sampling procedures, and finally encoded and stored into SDIF files. The spatial data can then be read by offline or real-time sound rendering software and adapted for different applications and system configurations.

5.1. Offline Rendering in OpenMusic: OM-Spat

OpenMusic is a privileged environment for offline generation and manipulation of spatial data related to formalized compositional models (see Section 2). A number of works have been carried out in this environment for the communication with sound processing tools, via external libraries or command line tools [3]. Nouno [10] also showed how this environment can be used to correlate spatialization data to the other parameters in a compositional process using the *OM-Spat* library [11], which first introduced the use matrices for the specification of spatial sound scenes.

We recently completed a major update of this library, introducing the new matrix and trajectory objects and the corresponding SDIF formatting and storage utilities presented in this paper (see Figure 1).

[3]This "single stream" solution could be used with non-synchronized sources as well by adding an "index" field in order to discriminate the different sources in the description matrices.

The SDIF files generated by *OM-Spat* can be interpreted by the *Spat* renderer,[4] a command line executable created with the Ircam Spat library [8]. The Spat renderer is called by OpenMusic and renders the spatial sound scene into multi-channel sound files for a given number or configuration of loudspeakers.[5]

5.2. Real-Time Rendering: Spat-SDIF-Player

In real-time environments the spatial data can be related to reactive programs in order to control sound rendering or other processes. We created a prototype player application in Max/MSP [13] (*Spat-SDIF-Player*, see Figure 7(a)) for the representation and real-time streaming of SDIF spatial description data.

Spat-SDIF-Player uses the *MuBu* multi-buffer container [14] in order to read the different frame streams in an SDIF file. It provides common transport controls for cueing and streaming this data as OSC messages, as well as an OpenGL 3D viewer for real-time visualization of the spatial sound scene.

The transmitted spatial description data are formatted in OSC following the SpatDIF specification [12] and can be received, interpreted and applied to sound sources or signal generators by any external spatial sound renderer or environment compliant with this protocol (Figure 7(b)).

6. CONCLUSION AND FUTURE WORKS

Our recent developments of compositional software for spatialization provide tools adapted both to the high-level, symbolic contexts of musical interaction and authoring, and to the lower-level, time-sampled representations required for spatial rendering. Using matrices as generic data structures in OpenMusic, we can represent large sets of sound scene descriptions, regardless of their complexity, and relate them to compositional processes.

SDIF seems to be a suitable format for subsequent writing and manipulation of these spatial descriptions. Structures such as streams, time-tagged frames, type descriptions, etc. constitute relevant intermediate entities and their generality ensures cross-compatibility with a large panel of sound processing tools. Flexible descriptions can be stored as asynchronous data streams of different rates and precisions, and then handled and interpreted in existing frameworks and musical environments supporting this format.

Communication and control of Spat 4 is currently operational from OpenMusic via the *OM-Spat* library or through OSC messages received and decoded in real-time. Some questions, however, still need to be addressed concerning the streaming and communication of spatial data.

[4]*Spat* renderer by Thibaut Carpentier, Ircam Acoustic and Cognitive Spaces team, distributed by Ircam.

[5]Other spatial rendering possibilities with Spat 4 include binaural synthesis or encoding using higher-order ambisonics. The descriptors used in *OM-Spat* are also compatible with *OMPrisma* [15] which can be employed for alternative rendering using other spatialization techniques (DBAP, ViMiC, etc.)

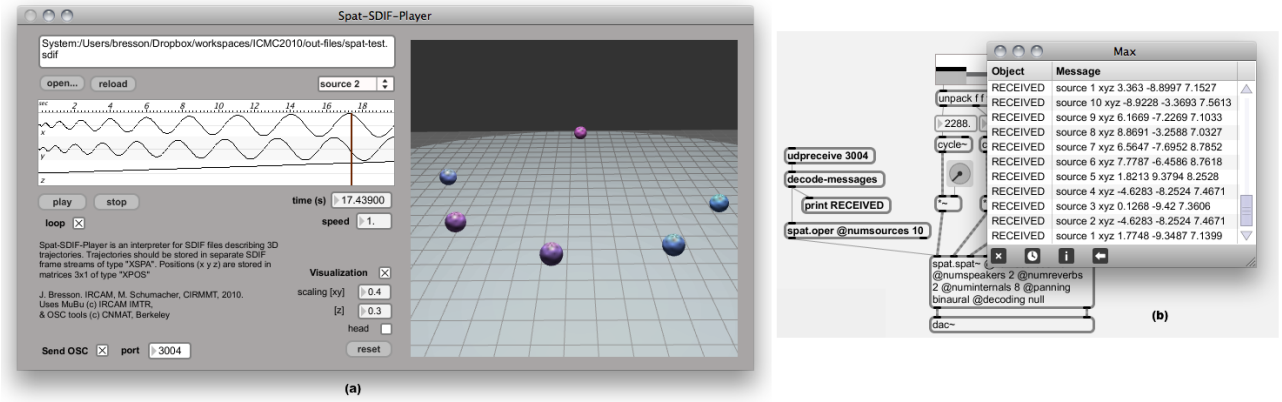

Figure 7. (a) Streaming and representation of SDIF-spatialization data with *SDIF-Spat-Player*. (b) Receiving and decoding data in Max/MSP for the control of Spat 4/SpatOper.

For instance the description of constant values, which are supposed to be sent once for all or at specific moments for instant updates, probably require different processing as compared to the continuously evolving parameters. Further works and specifications of storage and streaming protocols will probably help overcoming these issues.

7. REFERENCES

[1] C. Agon, M. Stroppa, and G. Assayag, "High Level Control of Sound Synthesis in OpenMusic," in *Proceedings of the International Computer Music Conference*, Berlin, Germany, 2000.

[2] G. Assayag, C. Rueda, M. Laurson, C. Agon, and O. Delerue, "Computer Assisted Composition at IRCAM: From PatchWork to OpenMusic," *Computer Music Journal*, vol. 23, no. 3, 1999.

[3] J. Bresson, "Sound Processing in OpenMusic," in *Proceedings of the International Conference on Digital Audio Effects*, Montréal, Canada, 2006.

[4] J. Bresson and C. Agon, "SDIF Sound Description Data Representation and Manipulation in Computer-Assisted Composition," in *Proceedings of the International Computer Music Conference*, Miami, USA, 2004.

[5] ——, "Musical Representation of Sound in Computer-aided Composition: A Visual Programming Framework," *Journal of New Music Research*, vol. 36, no. 4, 2007.

[6] J. Bresson, C. Agon, and M. Schumacher, "Représentation des données de contrôle pour la spatialisation dans OpenMusic," in *Actes des Journées d'Informatique Musicale*, Rennes, France, 2010.

[7] M. Geier and S. Spors, "ASDF: Audio Scene Description Format," in *Proceedings of the International Computer Music Conference*, Belfast, Ireland, 2008.

[8] J.-M. Jot and O. Warusfel, "A Real-Time Spatial Sound Processor for Music and Virtual Reality Applications," in *Proceedings of the International Computer Music Conference*, Banff, Canada, 1995.

[9] G. Kendal, N. Peters, and M. Geier, "Towards an Interchange Format for Spatial Audio Scenes," in *Proceedings of the International Computer Music Conference*, Belfast, Ireland, 2008.

[10] G. Nouno, "Some Considerations on Brian Ferneyhough's Musical Language through His Use of CAC. Part II — Spatialization as a Musical Parameter," in *The OM Composer's Book. 2*, J. Bresson, C. Agon, and G. Assayag, Eds. Delatour/IRCAM, 2008.

[11] G. Nouno and C. Agon, "Contrôle de la spatialisation comme paramètre musical," in *Journées d'Informatique Musicale*, Marseille, France, 2002.

[12] N. Peters, S. Ferguson, and S. McAdams, "Towards a Spatial Sound Description Interchange Format (SpatDIF)," *Canadian Acoustics*, vol. 35, no. 3, 2007.

[13] M. Puckette, "Combining Event and Signal Processing in the MAX Graphical Programming Environment," *Computer Music Journal*, vol. 15, no. 3, 1991.

[14] N. Schnell, A. Röbel, D. Schwarz, G. Peeters, and R. Borghesi, "MuBu & Friends - Assembling Tools for Content Based Real-Time Interactive Audio Processing in Max/MSP," in *Proceedings of the International Computer Music Conference*, Montreal, QC, Canada, 2009.

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

[16] 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.