A TECHNOLOGICAL PLATFORM FOR ANALYZING AND IMPROVING MUSICIANS' SOUND-GESTURE INTERACTIONS

Jocelyn Roze, Mitsuko Aramaki, Richard Kronland-Martinet, Sølvi Ystad

▶ **To cite this version:**

Jocelyn Roze, Mitsuko Aramaki, Richard Kronland-Martinet, Sølvi Ystad. A TECHNOLOGICAL PLATFORM FOR ANALYZING AND IMPROVING MUSICIANS' SOUND-GESTURE INTERACTIONS. e-Forum Acousticum 2020, Dec 2020, Lyon, France. 10.48465/fa.2020.0972 . hal-03099723

HAL Id: hal-03099723
https://hal.science/hal-03099723

Submitted on 6 Jan 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A TECHNOLOGICAL PLATFORM FOR ANALYZING AND IMPROVING MUSICIANS' SOUND-GESTURE INTERACTIONS

Jocelyn Rozé[1] **Mitsuko Aramaki**[1] **Richard Kronland-Martinet**[1] **Sølvi Ystad**[1]

[1] Aix-Marseille Univ, CNRS,
PRISM (Perception, Representations, Image, Sound, Music),
Marseille, 13009, France
roze@prism.cnrs.fr

ABSTRACT

This paper presents a technological platform aiming at analyzing and improving musicians' sound-gesture relationships. The conceptual foundations of the platform were driven by our latest research results, which highlight the close intertwining between sound quality and motor behavior among professional cellists. In particular, the results revealed that the cellists' timbre was consistently altered when they were deprived of their fine postural movements, such as torso swaying or head nodding. In a reciprocal way, we are now interested in investigating how subtle real time deformations of their natural timbre may affect their functional behavior. Besides the instrumental timbre, the platform should also enable to assess the musicians' motion/space intrications by modifying their surrounding acoustic space in an ecological way. After two years of development, we here present a mature architecture of our multimodal tool through the prism of these investigations. Such an architecture is particularly suitable for designing complex sound synthesis protocols involving the musicians' kinesphere perception related to the musical structure.

1. INTRODUCTION

When singing or playing an instrument such as the violin, expert musicians are trained to listen the sound produced several times for each note [1]. Such specialized skills of continuous auditory feedback exist because they were ingrained (or "embedded") in a concomitant way with the development of their sensorimotor perceptions [2, 3]. According to neuroscience, this strong correlation between the auditory and motor systems enables the mental construction of a single underlying musical representation [4, 5]. Experiments linked to musical chunking and coarticulation proposed to analyze these mental constructs as a composition of various sonic-gestural objects evolving at different scales along time [6, 7]. Such analyses in situations of expert musical performance provided very interesting insight in the way instrumentalists use their body in relation to the score structure and their expressive intention [8–11]. It also appears that the morphology of sonic-gestural objects strongly depends on space and timbre features : in the context of the cello for example, the instrumentalists tend to reduce their tempo in more reverberant rooms [12] and to produce "harsher" sounds when posturally impaired [13]. Such studies suggest that the sound percepts not only preexist in the musicians' body as timed motor actions, but also influence the motion execution as continuous temporal deformations.

In this paper, we present a technological tool that has been built to investigate these questions. This is a multimodal platform which makes it possible to manipulate the space and timbre features in real time, i.e. while the musicians are playing. Deformations of both sound facets can be applied in an ecologic manner at different metric scales of the musical score (note or phrase level). The motor effects caused by these acoustic transformations can be recorded, analyzed, and compared afterwards in a synchronous way. Regarding timbre manipulations, we synthesized a harshness phenomenon, which is a well-known timbre degradation in the case of bowed-string instruments. The characteristics of harsh sounds have been explored in depth in our previous studies on cello acoustics, and related to slower bow velocities, as well as insufficient postural rotations [14, 15]. We thus expect that professional cellists compensate unexpected harsh notes provided by our system through bow accelerations and wider movements or anatomic rotations. Regarding space manipulations, we applied existing spatialization technologies to simulate different room acoustics by means of multichannel impulse responses. The latter were obtained from recordings of several chapels performed in the context of the Sesames ANR project [1] .

The whole system aims at offering an integrated methodology to facilitate the development of experimental protocols assessing the influence of timbre and space acoustic features on human musical behavior.

2. ARCHITECTURE OVERVIEW

The conceptual foundations of the platform rely on a unified framework built for supporting a flexible wrapping and integration of different hardware and software solutions. Each new functionality must be implemented as a mod-

[1] "http://anr-sesames.map.cnrs.fr/

ule following precise rules, in order to be recognized and globally handled by the framework. This latter can then provide the modules with a set of common important services: status handling, data management, file access, networking communication and control protocols. The idea behind such an architecture is to enable a fast prototyping of new applications, with a quick-and-easy plugging of hardware devices or software components required by specific protocols of sound/gesture interactions. The modules controlled by the same protocol can be deployed on several machines in a transparent way thanks to the subjacent network layer. In the case of such a distributed configuration, the network layer also ensures the mappings to exchange data between the modules.

For fine musical purposes, a dedicated module provides the platform with a note-by-note protocol to trigger specific actions or acoustic events during the instrumental performance. These events can be captured and handled by all the modules connected to the framework. After a live performance, a global mechanism allows to compare and edit the multimodal features of one or several musicians. Regarding the association between the software framework and our experimental hardware setup, we can basically separate 4 main components: A DMI (Digital Musical Instrument) and three kinds of processing chains : audio, motion and sensor modules.

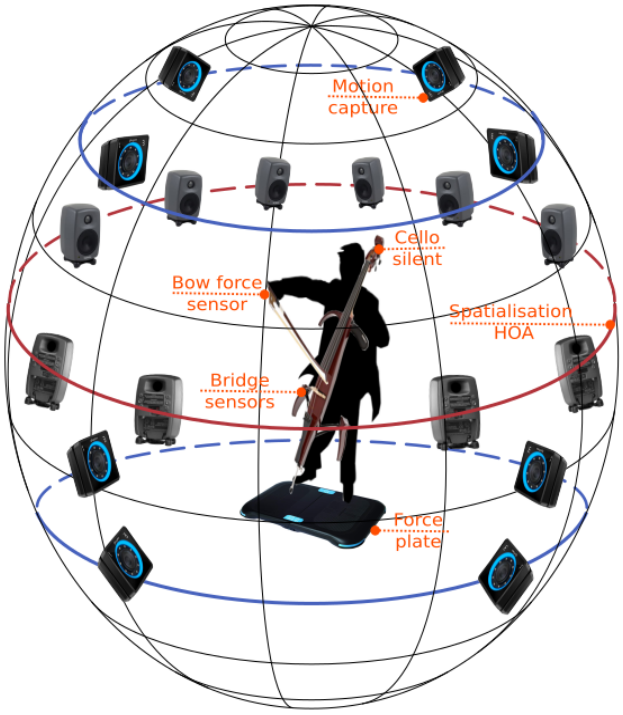

Figure 1. Aparatus overview. The bow/string interaction signals are collected through pickups mounted on the cello bridge. Both signals are then processed by the acoustic chain and rendered in real time within a configurable network of loudspeakers. Cellist's movements are collected by a motion capture system. The bow force and support transfers between the feet are estimated by a force sensor and a force plate respectively.

2.1 Digital Musical Instrument

Our DMI is based on an electric silent cello (Yamaha Silent SVC 210 [2]). The silent feature was required to minimize the direct acoustic feedbacks with respect to the sound rendered from our acoustical transformations. In spite of a robust conception and accurate ergonomics, the sound delivered by the internal circuitry of this instrument didn't bring us satisfaction for a fine work on the timbre quality. We therefore adopted a solution based on two contact pickups mounted on each side of the bridge : a contact microphone (AKG-C411 [3]) to rebuild the perception of an acoustic cello, and a piezoelectric sensor (IMELOD [4]) to introduce timbre deformations (see below).

2.2 Audio processing chain

Our audio chain is based on a fast Gigabyte audionumeric network communicating through the Dante protocol [5] . An *audio input module* (cf section 3.2) collects the two vibration signals from the bridge pickups. The signal provided by the contact microphone is processed by a *convolver module* (cf section 3.3) which performs its convolution in real-time by a pre-computed impulse response of the acoustic body of the cello. Two criteria determined the choice of the AKG-C411 microphone as the source signal for this convolution: a very flat frequency response whatever be the cello registers, and a good resistance to electromagnetic interferences.

In the case of timbre manipulations, a *harshness module* (cf section 3.5) processes both the "fake" (modified) cello signal obtained by convolution, and the raw signal from the piezoelectric sensor. The latter filters much more low-frequency components than the contact microphone, which is ideal to simulate a perceptual harsh sensation, i.e. of a sound "lacking of matter, without depth and natural resonance" [13]. From a phenomenological point of view, this actually may be related to an instrument deprived of its resonating body, only reduced to the raw bow/string interaction.

In case of spatial manipulations, a *spatializer module* (cf section 3.6) processes only the "fake" cello signal by convolving it to specific room acoustic responses. The signal is first encoded into a HOA (High Order Ambisonic) basis and then decoded after the multichannel convolution step for rendering on a set of target loudspeakers. To enable a full acoustic immersion of the musician, we here use a spherical setup of 42 loudspeakers (cf Figure 1) located in an anechoic room of our laboratory. This 4-meter diameter metallic structure can be used to simulate tridimensional sound landscapes provided that the head of the subjects (here the cellists) is located in the center of the sphere [16].

[2] "https://fr.yamaha.com/fr/products/musical˙instruments/strings/silent˙series/svc210/index.html
[3] "https://www.akg.com/Microphones/Condenser Microphones/C411PP.html
[4] "http://www.imelod.com/en/products˙view.asp?id=760
[5] "https://www.audinate.com/meet-dante/what-is-dante

2.3 Motion processing chain

Our motion processing chain is based on an Optitrack motion capture system of eight infrared cameras [6], which are regularly laid out onto the spherical structure (cf Figure 1). A *streaming module* (cf section 3.7) collects the tridimensional coordinates of reflective markers provided by the system. The configuration of the markersets proposed by the module is adjustable and can dynamically match the streaming of rigid bodies (non-deformable objects), skeletons (human body templates) or a combination of both. For our application, we streamed two rigid objects - the electric cello (5 markers) and the bow (3 markers) - as well as a template of legacy upper body (25 markers).

The streamed motion data can then be processed in various ways depending on the purposes : a *kinematic3D module* computes the instantaneous velocity and acceleration of one or more selected markers with varied smoothness. An *angular3D module* computes the angles between a selection of rigid bodies or anatomic body segments. A *PCA module* performs Principal Component Analysis from a prerecorded sequence of streamed markers. Finally, a *gesturefollower module* was also built to regulate the proportion of harshness feedback according to the percentage of motion compensation expected by our hypotheses. But we finally kept apart this possibility by considering that it might cause a severe experimental bias regarding unexpected cellists' behavioral reactions.

2.4 Sensor processing chain

Our sensor processing chain aims at collecting the signals of various sensors from specific wired electronic equipments, or wireless transmitted through bluetooth or wifi technologies (cf Figure 1). For our application, a *bowforce module* has been designed to get an estimation of the hair deflexion applied by the bow on the cello strings. This force signal is measured by a small FSR sensor mounted inside the bow frog, numerically encoded and sent over the wifi network of the laboratory by an electronic card [7] embedded on the cellists' thigh. A *forceplate module* has also been implemented to estimate the support transfers applied by the cellists' feet on the ground. As highlighted by recent studies [17], the commercial wii balance board [8] turned out to provide quite good estimations of the COP (Center Of Pressure) displacements, although we had to scale the measures to take into account the feet pressure only. When combined together, the bowforce and forceplate modules allow to assess the so-called "quadrilateral transfer" [18] of energy circulating from the cellists' feet to the bow.

3. TECHNICAL IMPLEMENTATIONS

The whole framework has been developed within the Max/MSP environment to satisfy real-time constraints. We here describe the main technical features of the modules composing this platform, and justify our implementation choices as far as possible.

3.1 Common features

As presented in the architecture overview, each module embeds a set of common features ensuring their homogenous integration inside the platform. We paid special attention to provide the different kinds of components (audio, motion, sensor) with synchronized processing capabilities in real or non real time. The MUBU container technology [19] turned out to be very suitable for these needs, because it proposes powerful features of dynamic track configuration and thread-safe data access. We thus embedded a MUBU container to each module of our platform as a common but customizable feature of data storage/retrieval. More precisely, a module container can be accessed according to 2 modes:

- a *LIVE* mode used in real-time situation, i.e. during the performance. This mode offers essential functionalities, such as data recording, replaying, scrubbing and file storage/retrieval in the appropriate format (WAV files for audio modules, SDIF files for motion and sensor modules).
- an *EDIT* mode used in off-line situations, i.e. after the performance. This mode should propose flexible data visualizations, such as comparing a set prerecorded data sequences, focusing on some markers of a large motion capture dataset, or zooming in the multimodal data range of a particular note.

The whole synchronization process between the modules is ensured by a dedicated network layer, which has been developed in Javascript/NodeJS and integrated in the Max/MSP environment through the Node4Max engine [9]. A pilot module plays the role of conductor and embeds the server part of the network engine. All other modules are considered as clients, and automatically attempt to connect to a pilot during their starting phase to be remotely controlled. If they can access a pilot, they also receive protocol label to dynamically perform an initialization with parameters adapted to a given experimental context (cf section 3.8). Otherwise, they initialize with default parameters and work in a standalone mode. Regarding the close ways to handle both network and preset data, we decided to adopt a single mean of data exchange by OSC messages through the whole platform. The ODOT [20] technology turned out to be very suitable for this need, and also flexible enough to perform standard algorithmic operations during the transfer of OSC bundles. Functionally, a module can deal with 2 kinds of OSC message to communicate with its environment:

- *Status* messages are used to load presets of parameters and to monitor the internal state of the module. Any parameter change in the module GUI is reported into its status bundle. The latter can then be written on the disk into a formatted JSON status file.

[6] "https://optitrack.com/cameras/primex-13w/
[7] "https://interface-z.fr/pronfiture/sans-fil/252-1750-miniwi.html
[8] "https://fr.wikipedia.org/wiki/Wii˙Balance˙Board
[9] "https://docs.cycling74.com/nodeformax/api/

- *Data* messages are used in every module processing task and to exchange information with the network layer or any external device. For example, a motion capture data stream can be captured on one machine and forwarded to another machine for sound spatialization operations.

Given its hierarchical nature, this based-OSC messaging mechanism enables a precise structuration of any information processed by the modules. Regarding the status messages, we affect a root OSC namespace to each module, and build the functionalities it can propose as a hierarchy of dedicated OSC subspaces. This rule implies that the module parameters must be defined and ordered within their functional OSC subspace. For most of the modules, we made the choice to expose the graphical parameters of a given functionality into a dedicated view, which can be selected from a menu at the left-upper side of its module (cf examples for each module below). Regarding the data messages, we classify the OSC namespace according to the type of their native processing chain (*/audio*, */motion* and */sensor*). A fourth type (*/control*) also exists for the messages produced by the pilot module. Such a full OSC architecture turned out to be very efficient for ensuring most of the required synchronization tasks (recording, playing, scrubbing..) in a distributed environment.

3.2 Audio input

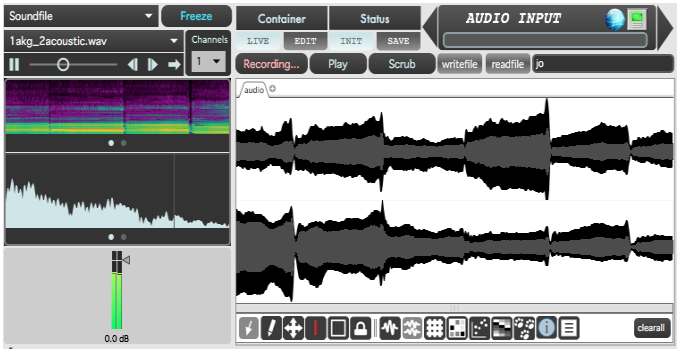

Figure 2. The audio input module.

The audio input module (alias */audioinput*) proposes two main functionalities :

- A sound file player (subspace */sndfile*). This view is adjustable to different channels of the sound file.
- A sound card reader (subspace */sndcard*). This view is also adjustable to different channels of the sound card.

A button allows to freeze the stream of samples delivered by the sound file player or the sound card reader. This functionality is achieved with the *mubu.granular~* object, which can also be dynamically configured to a different number of input channels. The period and duration of the granulation process can be changed in the status file of the module.

3.3 Audio convolver

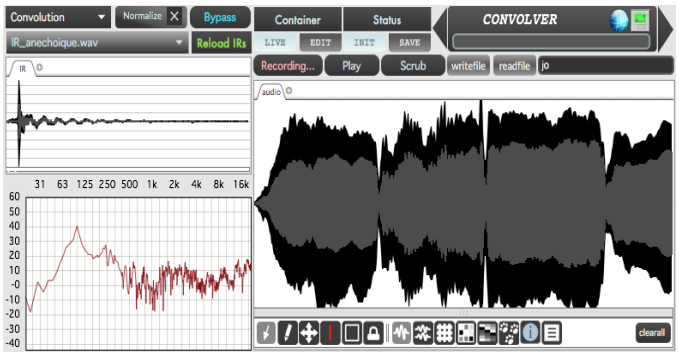

Figure 3. The audio convolver module.

The audio convolver module (alias */convolver*) proposes two main functionalities:

- A convolution by a monochannel impulse response (subspace */convolution*). This view displays both current selected impulse response and its frequency response. The convolution process is performed with the *multiconvolve~* object of the HISS Toolbox [21].
- A deconvolution for computing a monochannel impulse response (subspace */deconvolution*). This view requires two synchronized input channels and computes their transfer function with the *irreference~* object of the same HISS toolbox.

To simulate the perception of an acoustic cello from our silent electric instrument, we performed the deconvolution between two pickup signals : an AKG-C411 microphone mounted on the bridge of an acoustic instrument, and a near-field NEUMANN microphone located one meter from the strings. According to the literature [22], we performed a synchronized measure of these two signals by playing a three-octave scale in an anechoic room, before applying the deconvolution process. A button in the convolution view allows to reload the full set of impulse responses which can be applied to the input signal. The convolution process can also be bypassed, which is particularly useful for comparing the perceptual acoustic result with and without convolution.

3.4 Audio score follower

The audio score follower module (alias */scorefollower*) proposes three main functionalities:

- Following an audio signal (subspace */audio*).
- Following midi events (subspace */midi*).
- Following pitch data (subspace */pitch*).

Each of these functionalities relies on a same external, *antescofo~* [23], which implements very efficient algorithms for anticipatory score following. If the pitch and midi configuration are interesting for testing purposes, we

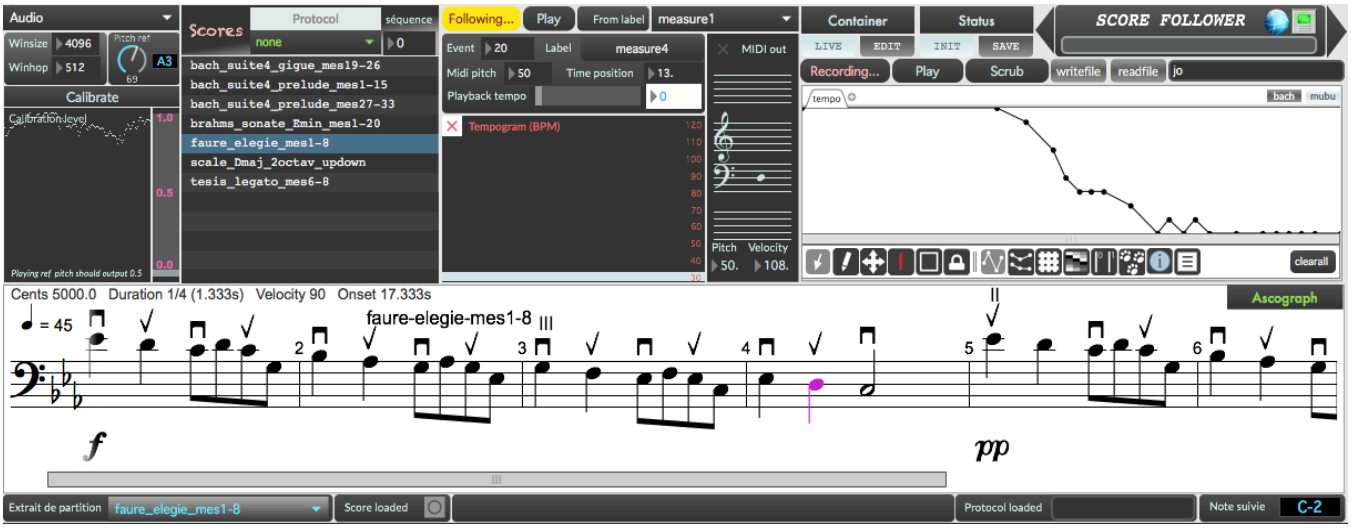

Figure 4. The score follower module.

mainly used Antescofo in the audio tracking mode. The corresponding view of our module allows to calibrate Antescofo with default parameters, which are suitable for covering the cello tessiture: a window size of 4096 samples and an overlap hop size of 512 samples were used for this purpose. The reference pitch note could also be kept on an A3. Note that a special attention should be paid to the nature of the input signal provided to Antescofo : our preliminary tests with piezoelectric sensors on the bridge resulted in a poor note detection. More accurate tracking was ensured with the audio signal produced by the AKG microphone, probably because of a better immunity to electromagnetic interferences.

Before any score following operation, a score excerpt must first be loaded into the Antescofo engine. Our module proposes a menu to select these excerpts and load them in the Antescofo native format (TXT). Each excerpt is also associated to a file in the MusicXML format to convert the musical representation into a score of notes on loading. This visual functionality is ensured by a specialized object called *bach.score*$\sim$ [24]. Furthermore, an excerpt can be remotely selected by the pilot module if executed according to a given protocol. In this case, the score excerpt is attached to a unique sequence identifier which characterizes its location in the protocol scenario. The declaration of this protocol in the module should be preliminary achieved with the "Protocol" button, and saved in a JSON file on the disk. In fact, the main purpose of such a protocol definition is to associate each note of the musical score with specific control actions. Consequently, the JSON protocol file has always 3 sections: scores, actions, and sequences. The latter links the scores to the actions according to unique combinations.

Our module has a panel to launch Antescofo in a playing or following mode. The playing mode allows to execute the score in a metronomic way without any note detection. The following mode reports any new note transition during a live performance, by providing its midi informations (pitch/velocity) as well as a local tempo estimation for each detected note. The latter information is very useful to reveal the musicians' perception of time with respect to our experimental factors (timbre and space alterations). Tempo variations are continuously recorded into the module container during the score following process, and a midi file of the performance is generated when it stops. Furthermore, we implemented some useful live functions such as selecting the current note detected by Antescofo in the *bach.score*$\sim$ object, and sending its corresponding action in the protocol to any listening module in the network. These actions can then be captured by the two main modules - i.e. harshness and spatialization - of our platform.

3.5 Audio harshness

In our analysis of the harshness phenomenon [13], we could essentially link its perception to two kinds of signal features: the emergence of formantic areas and a more chaotic behavior of the partials along time, also called harmonic asynchrony. The harshness module (alias */harshness*) is thus implemented as a meta-module composed of two submodules likely to synthesize these functionalities:

- A submodule of the LPC morphing (subspace */morphinglpc*). This component controls the proportion of formant applied to the input signal, which should perceptually result in a more or less shrill sound.
- A submodule of harmonic modulation (subspace */harmodulator*). This component controls the proportion of harmonic asynchrony applied to the signal which formant has been modified. This process should perceptually result in a more or less unstable Helmholtz motion.

The control messages coming from the score follower are redirected by the harshness module to these two components. For now, the harshness protocol defines 4 kinds of action identifiers: "0:not harsh; 1:semi formant; 2:full formant; 3:fully harsh (i.e. full formant and asynchrony)". As explained in section 3.4, each one of these actions can

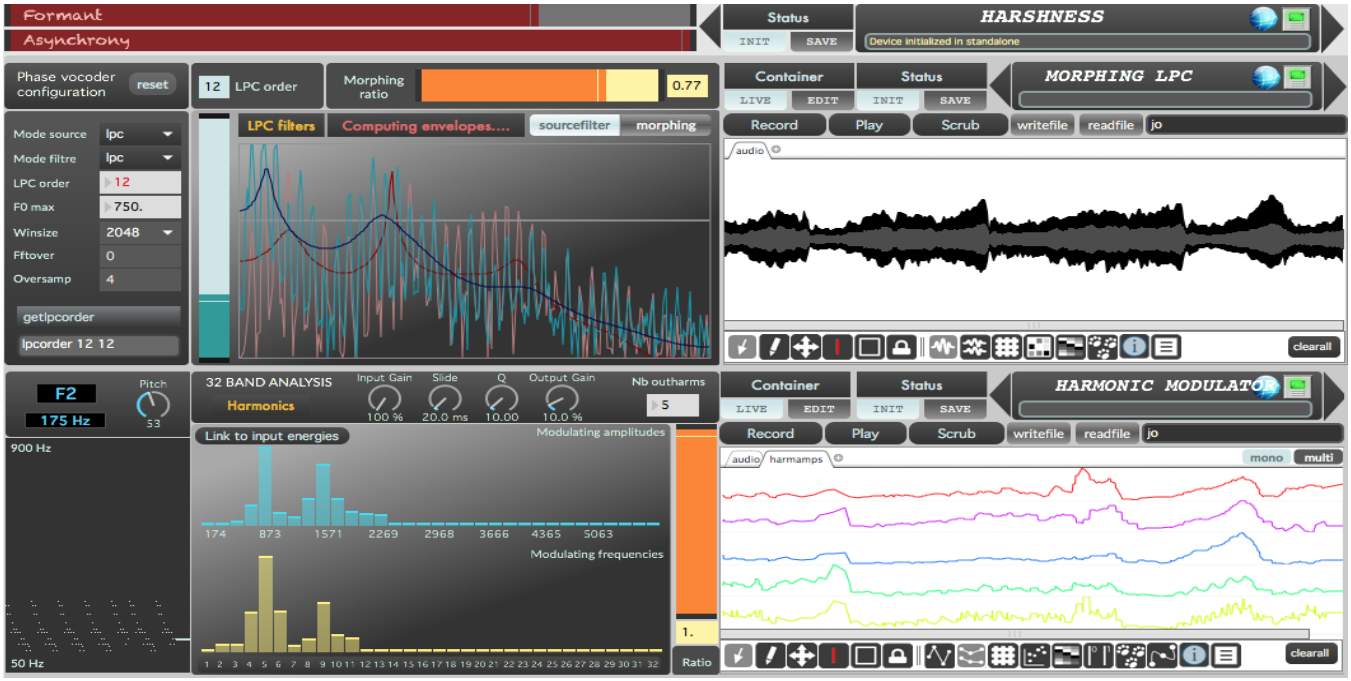

Figure 5. The harshness module.

be associated to specific notes of a score excerpt. They will be triggered by these notes during the score following process and will alter the "fake" cello sound in real-time accordingly.

3.5.1 Audio morphing LPC

This submodule performs an adjustable LPC cross-synthesis between two input signals by means of a source/filter transformation of their spectral envelopes [25]. As explained in section 2.2, such a technique has the potential to generate a formant in an ecological (and phenomenological) way if we apply the cross-synthesis between two pickup signals capturing the same instrumental source (the cello bridge), but characterized by different frequency responses. In our case, LPC cross-synthesis is achieved from a source signal corresponding to the "fake" cello sound built from the AKG pickup (flat frequency response), and a filter signal corresponding to the raw sound of the piezolelectric sensor (highpass frequency response). The process results in a signal with boosted high frequency contents which sounds like an acoustic instrument played in a shrill and unpleasant way.

The submodule GUI provides a view to tune the parameters of the source/filter algorithm, such as the number of LPC coefficients. A dedicated panel also provides a visualization of the spectral envelopes for the source and filter signals, by resynthesizing them from their LPC coefficients. We chose to model both signals' spectral envelopes with a number of 12 LPC coefficients, which turned out to be sufficient for fine perceptual degradations. The algorithm was preliminary implemented by transposing its Matlab code in real-time with the tools of the FTM/GBR distribution [26]. However, this adaptation wasn't fully satisfactory, since unpredictable audio artifacts occurred during the filtering of the whitened source signal. We therefore opted for the *supervp.sourcefilter~* external object [10] to perform the task with an analysis window of 1024 samples overlapped by a factor 4. Such a configuration enabled high-quality morphings between different proportions of harsh sounds continuously applied to the source signal.

3.5.2 Audio harmonic modulator

This submodule first performs the harmonic analysis of an input signal by decomposing it into 32 energy sub-bands. To this aim we first used the *fffb~* object, which implements an efficient bank of bandpass filters, and offers several controls for adjusting their respective central frequencies, gain levels and selectivity factors. We then synthesized perceptual proportions of harmonic asynchrony by modulating each energy band of the filterbank with oscillators of specific frequency/amplitude. These parameter pairs have been chosen according to the results of our harshness analyses [13], which highlighted the emergence of main formantic areas around the 5^{th} and 9^{th} harmonics.

The submodule GUI displays the current fundamental frequency from which the 32 harmonic filter bands are computed. This information should be continuously tracked along the performance in order to keep the filterbank correctly tuned. The platform provides two ways of getting this pitch information: through a dedicated pitch tracking module (based on *fzero~* or *pipo.psy~* algorithms); through the MIDI informations delivered by the score following module. A central panel allows to configure the frequency and amplitude parameters for each oscillator associated to the filterbank channels. It also contains a button for linking the amplitude modulations of each os-

[10] "https://forum.ircam.fr/projects/detail/supervp-trax/

illator to the amount of input energy estimated for each channel. This function turned out to be perceptually efficient for increasing the harmonic asynchrony of harsh sounds while preserving their naturalness. Finally, another useful function enables to record only a subset of the harmonic amplitude laws into the module container. We here limited the recordings to the five first harmonics to avoid CPU overload.

3.6 Audio spatializer

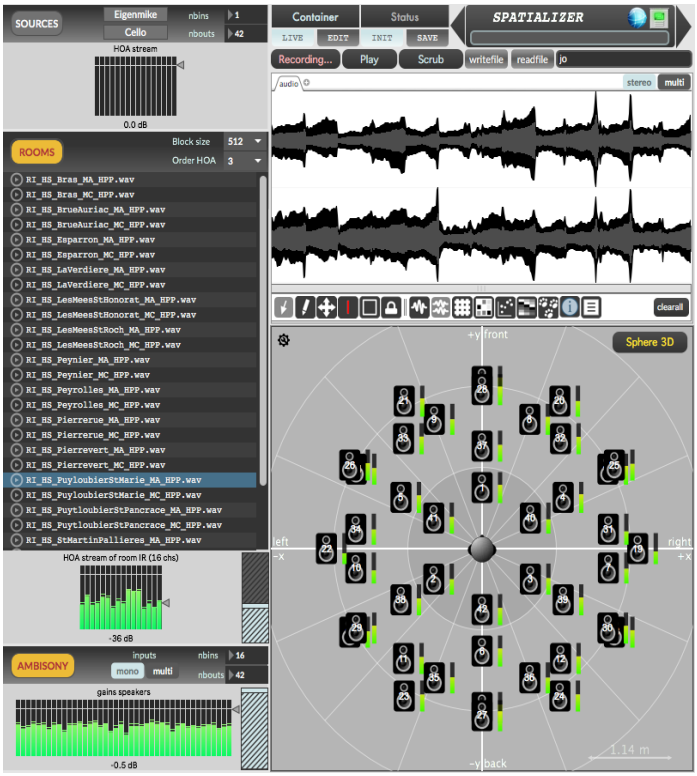

Figure 6. The spatializer module.

The audio spatializer module (alias */spatializer*) proposes three main functionalities embedding a set of externals of the SPAT software [27]:

- Handling of sound sources (subspace */spatsource*). This panel automatically displays the number of sound sources which coexist within our spherical space. The number of audio input channels should coincide with the number of spatial sources (generally streamed by motion capture). These audio sources are encoded on a HOA basis of spherical harmonics by the *spat5.hoa.encoder~* object.
- Handling of room responses (subspace */spatroom*). This panel allows to select a 32-channel room response, and to convolve it with an input signal in real-time. These spatial room responses are precomputed from recordings performed in several chapels with a 32-capsule Eigenmike [28]. They are also pre-encoded at 4^{th} HOA order, i.e. on a basis of 25 spherical harmonics. The multichannel convolution process is carried out by the *spat5.conv~* object.
- Handling of ambisonic rendering (subspace */spatambi*). This panel performs the rendering of an HOA-encoded audio stream onto different loudspeaker configurations. The *spat5.hoa.decoder~* can be dynamically configured to decode an HOA stream on setups of 9, 16, 32 until 42 speakers. For each change of the output channel setup, the routing table of our digital audio system should be updated to correctly map them to specific loudspeakers.

A button allows to display a 3D view of the loudspeaker setup as well as the source locations actually streamed by the system. We used the *Dante Virtual Soundcard* and *Dante Controller* softwares [11] to configure the routings between the output audio channels of our module and the target loudspeakers. Our spherical layout of loudspeakers turned out to be suitable for simulating perceptual immersions of the cellists within different room acoustics, provided that their head remained in the "sweet spot", i.e. in the center of the structure. Taking into account our hardware limitations, we got fine acoustic renderings for live performances until a 3^{rd} HOA order on the full set of 42 loudspeakers. The module also offers the possibility to record all audio output channels or a stereo mix to spare the CPU resources.

3.7 Motion capture streaming

The module for streaming motion capture data (alias */streaming*) proposes a view to plot the marker coordinates, that automatically adapts to the currently streamed Optitrack model. We developed a customized Python program to convert the binary frames streamed by *Motive* (the Optitrack software [12]) into readable OSC messages. The Python program then forwards these OSC messages to our module according to a common data formatting. Hence, each marker model that can be streamed by Motive should be declared in a specific JSON template file accordingly. The information contained in these template files also enables the reconstruction of a model avatar, which can be displayed in real-time through a dedicated GUI button.

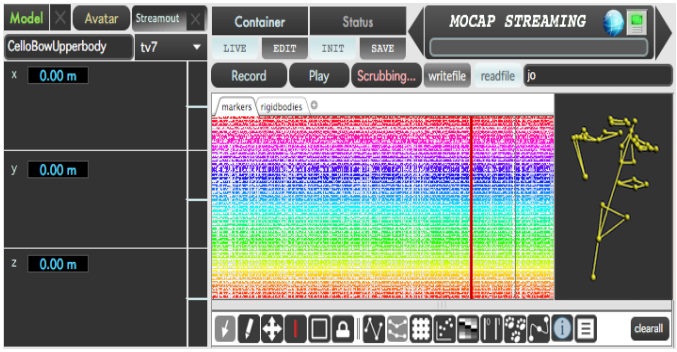

Figure 7. The mocap streaming module.

Figure 8. The pilot module.

3.8 Pilot

The pilot module (alias */pilot*) proposes two main functionalities:

- Handling a database of subjects (subspace */database*). This database defines the protocols that will be applied to the subjects, and their performance scenario, i.e. their individual processing order for the takes planned by a protocol. Once a protocol has been selected, a panel allows to create or edit the subjects with their performance scenario. The selection of a given sequence of the scenario is achieved by a button entitled "Next take". This sequence identifier can then usually be used to select a particular musical excerpt in the score following module (cf section 3.4). The subjects' database can also be accessed in a live or edit mode, in order to globally communicate with the client module containers in the appropriate mode.

- Handling the setup of synchronized modules. On the loading of a protocol, the pilot waits for the connection of a given set of client modules. When an expected module connects, a bidirectionnal TCP network link is established through the *websockets* technology [13] . Actually, this works as a chat forum where clients can subscribe to specific topics (here referenced as protocol labels), and get synchronized control messages from the server or from specific clients. The pilot presets define for each protocol the list of modules which are authorized to connect, as well as the network mapping rules to forward messages between specific modules. Once a protocol has been loaded, the list of expected modules with their connection states can be visualized by clicking the little globe-shaped button.

The pilot bottom pane displays a timeline, which provides the timetag currently shared by each module connected to the network. This information is crucial for the correct synchronization of global operations (recording, playing, scrubbing, write/read a take file...). A dedicated slider allows a global scrubbing control for all the connected module containers. Above, the timetag is both presented as absolute data (in min/sec/ms) and a tempo-related data (in bar/beat/units) depending on the current metric signature ($4/4$ by default). The latter formatting is ensured by a named *transport* object (the global Max transport should be avoided if the scorefollower module is connected). The protocols defined in the pilot database can also provide metadata to adjust the tempo/signature used during the performances, as well as the working parameters of the set of connected modules.

[11] "https://www.audinate.com/products/software/dante-controller

[12] "https://www.optitrack.com/software/motive/

[13] "https://fr.wikipedia.org/wiki/WebSocket

4. CONCLUSION

In this article, we presented a technological platform which aims to analyze and improve the musicians' sound-gesture interactions. It is designed as a unified framework of multimodal components, that can be globally controlled through efficient synchronization mechanisms. A special attention has been paid to preserve the quality of interactions between the human perception (learned with a mechanic instrument) and the sound synthesis processes (associated with an electric instrument). From a phenomenological point of view, this approach is developed to subtly distort the naturalness of instrumental feedbacks and assess human adaptations accordingly. The results expected from these experiments may provide valuable insights in audio-motor behavior among musicians and provide clues to establish new sonification strategies based on time-space dependencies, which are likely to improve the musicians' kinesphere perception.

5. REFERENCES

[1] X. Gagnepain, *Du musicien en général... au violoncelliste en particulier*. Cité de la musique, 2001.

[2] M. Leman, "The role of embodiment in the perception of music," *Empirical Musicology Review*, vol. 9, no. 3-4, pp. 236–246, 2014.

[3] P.-J. Maes, M. Leman, C. Palmer, and M. M. Wanderley, "Action-based effects on music perception," *Frontiers in psychology*, vol. 4, p. 1008, 2014.

[4] R. J. Zatorre, J. L. Chen, and V. B. Penhune, "When the brain plays music: auditory–motor interactions in mu-

sic perception and production," *Nature Reviews Neuroscience*, vol. 8, no. 7, pp. 547–558, 2007.

[5] I. Wollman, V. Penhune, M. Segado, T. Carpentier, and R. J. Zatorre, "Neural network retuning and neural predictors of learning success associated with cello training," *Proceedings of the National Academy of Sciences*, vol. 115, no. 26, pp. E6056–E6064, 2018.

[6] R. I. Godøy, A. R. Jensenius, and K. Nymoen, "Chunking in music by coarticulation," *Acta Acustica united with Acustica*, vol. 96, no. 4, pp. 690–700, 2010.

[7] A. R. Jensenius, M. M. Wanderley, R. I. Godøy, and M. Leman, "Musical gestures : Concepts and methods in research," *Musical gestures: Sound, movement, and meaning*, vol. 12, pp. 12–35, 2009.

[8] M. M. Wanderley, B. W. Vines, N. Middleton, C. McKay, and W. Hatch, "The musical significance of clarinetists' ancillary gestures: An exploration of the field," *Journal of New Music Research*, vol. 34, no. 1, pp. 97–113, 2005.

[9] F. Visi, E. Coorevits, E. Miranda, and M. Leman, *Effects of different bow stroke styles on body movements of a viola player: an exploratory study*. Ann Arbor, MI: Michigan Publishing, University of Michigan Library, 2014.

[10] T. F. Santos and M. A. Loureiro, "The observation of ancillary gestures using the laban-bartenieff movement fundamentals," in *Anais do XIII Simpósio Internacional de Cognição e Artes Musicais*, 2017.

[11] C. Massie-Laberge, I. Cossette, and M. M. Wanderley, "Kinematic analysis of pianists' expressive performances of romantic excerpts: Applications for enhanced pedagogical approaches," *Frontiers in psychology*, vol. 9, p. 2725, 2019.

[12] Z. Schärer Kalkandjiev and S. Weinzierl, "The influence of room acoustics on solo music performance: An empirical case study," *Acta Acustica united Ac*, vol. 99, no. 3, pp. 433–441, 2013.

[13] J. Rozé, M. Aramaki, R. Kronland-Martinet, and S. Ystad, "Exploring the perceived harshness of cello sounds by morphing and synthesis techniques," *The Journal of the Acoustical Society of America*, vol. 141, no. 3, pp. 2121–2136, 2017.

[14] J. Rozé, M. Aramaki, R. Kronland-Martinet, and S. Ystad, "Assessing the effects of a primary control impairment on the cellists' bowing gesture inducing harsh sounds," *IEEE Access*, vol. 6, pp. 43683–43695, 2018.

[15] J. Rozé, M. Aramaki, R. Kronland-Martinet, and S. Ystad, "Cellists' sound quality is shaped by their primary postural behavior," *Scientific reports*, vol. 10, no. 1, pp. 1–17, 2020.

[16] G. Parsehian, L. Gandemer, C. Bourdin, and R. K. Martinet, "Design and perceptual evaluation of a fully immersive three-dimensional sound spatialization system," in *3rd International Conference on Spatial Audio (ICSA 2015)*, 2015.

[17] M. Thirumalai, W. B. Kirkland, S. R. Misko, S. Padalabalanarayanan, and L. A. Malone, "Adapting the wii fit balance board to enable active video game play by wheelchair users: User-centered design and usability evaluation," *JMIR rehabilitation and assistive technologies*, vol. 5, no. 1, p. e2, 2018.

[18] P. De Alcantara, *Technique Alexander pour les musiciens*. Editions AleXitère, 2000.

[19] N. Schnell, A. Röbel, D. Schwarz, G. Peeters, R. Borghesi, *et al.*, "Mubu and friends–assembling tools for content based real-time interactive audio processing in max/msp," in *ICMC*, 2009.

[20] A. Freed, J. MacCallum, and A. Schmeder, "Dynamic, instance-based, object-oriented programming in max/msp using open sound control message delegation.," in *ICMC*, 2011.

[21] A. Harker and P. A. Tremblay, "The hisstools impulse response toolbox: Convolution for the masses," in *Proceedings of the international computer music conference*, pp. 148–155, The International Computer Music Association, 2012.

[22] A. Pérez Carrillo, J. Bonada, J. Pätynen, and V. Välimäki, "Method for measuring violin sound radiation based on bowed glissandi and its application to sound synthesis," *The Journal of the Acoustical Society of America*, vol. 130, no. 2, pp. 1020–1029, 2011.

[23] A. Cont, "Antescofo: Anticipatory synchronization and control of interactive parameters in computer music.," in *International Computer Music Conference (ICMC)*, pp. 33–40, 2008.

[24] A. Agostini and D. Ghisi, "Bach: An environment for computer-aided composition in max," in *ICMC*, 2012.

[25] U. Zölzer, *DAFX: digital audio effects*. John Wiley & Sons, 2011.

[26] N. Schnell and D. Schwarz, "Gabor, multi-representation real-time analysis/synthesis," in *Proc. of the 8 th Int. Conference on Digital Audio Effects*, 2005.

[27] T. Carpentier, M. Noisternig, and O. Warusfel, "Twenty years of ircam spat: looking back, looking forward," in *Proc of 41st International Computer Music Conference (ICMC)*, pp. 270 – 277, 2015.

[28] J. Meyer and G. Elko, "A highly scalable spherical microphone array based on an orthonormal decomposition of the soundfield," in *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. II–1781, IEEE, 2002.