Audio-based Musical Artificial Intelligence and Audio-Reactive Visual Agents in Revive

Kıvanç Tatar
Simon Fraser University
ktatar@sfu.ca

Philippe Pasquier
Simon Fraser University
pasquier@sfu.ca

Remy Siu
Simon Fraser University
remy@remysiu.com

ABSTRACT

Revive is an live audio-visual performance project that brings together a musical artificial intelligence architecture, human electronic musicians, and audio-reactive visual agents in a complex multimedia environment of a dome view with multichannel 3D audio. The context of the project is live audio-visual performance of experimental electronic music through structured improvisation. Revive applies structured improvisation using cues and automatized parameter changes within these cues. Performers have different roles within the musical structures initiated by the cues. These roles change as the performance temporally evolves. Sonic actions of performers are further emphasized by audio-reactive visual agents. The behaviours and contents of sonic and visual agents change as the performance unfolds.

1. INTRODUCTION

Revive is a live audio-visual performance project that features two human performers, and MASOM, which is a musical agent, an artificial intelligence (AI) architecture for live performance [1]. For each sonic performer in Revive, a corresponding visual agent puts sonic gestures and textures into live generative images. The visual agents use a machine listening algorithm in the input module to exhibit audio-reactive behaviours with generative visuals. This reveals the musical gestures that are so often lost in electronic music performance.

Revive's aesthetics span a variety of experimental electronic music styles including acousmatic music, soundscapes, glitch, intelligent dance music (IDM), and noise music. Acousmatic compositions use electronic means to create or process sounds to produce compositions. Soundscapes use field recordings of the environment outside the studio as the musical content. Glitch music explores the idea of using sounds that are generated by the failure of any procedure. For example, glitch performers overload the cpu to generate clicks and drops on the audio output. IDM composers use any sound object to produce dance music, extending their audio palette to unconventional sounds. Glitch sounds such as clicks, short impulsive noises frequently appear in IDM compositions. Noise music stands on the louder and aggressive end of the musical composition continuum. Noise music employs loud sounds to stimulate the body. The stimulations can be an ear pain caused by the loud sounds or pulsations generated by loud bass frequencies to vibrate the human body.

[1] A recording of a *Revive* session is available at https://kivanctatar.com/revive where the audio is the binaural encoding of the 3D audio setup.

The musical AI in *Revive*, Musical Agent based on Self-Organizing Maps (MASOM) is an audio-based musical agent with autonomous unsupervised learning. Musical agents are artificial agents that automatize musical creative tasks [1]. Musical agents differ from purely generative systems because of autonomy, reactivity, proactivity, adaptability, coordination, and emergence behaviours.

The architecture of MASOM proposes an innovative approach by combining a sonic latent space generation with statistical sequence modelling for temporal musical structure. The autonomous, unsupervised learning in MASOM only requires audio recordings. The musical agent creates a sound memory through automatic audio segmentation and thumbnailing, using audio features of timbre, loudness, fundamental frequency, duration, and music emotion features of eventfulness and pleasantness. The agent organizes sounds on a two-dimensional map so that similar sound clusters locate closer to each other. MASOM learns the temporality of musical form by applying statistical sequence modelling on the organized sound memory. Hence, the agent assumes the musical form as temporal shifts on sound clusters that are organized in the feature space. In *Revive*, we use a variation of MASOM architecture, where previous statistical sequence modelling algorithm, VMM-PPM-C is exchanged with the Factor Oracle algorithm [14] to improve timbre consistency for the cases where the agent is trained on big-size audio recording data ranging from 1-GB to 100GB.

Revive project exemplifies how to incorporate an unsupervised, audio-based musical AI system into an audio-visual live performance. The cue system of Revive automatizes parameter changes that initiate musical sections in the performance. This allows performers to focus more on the aesthetics and less on the technical complexities. The cue system defines roles where the performers can explore improvisation within musical roles. Some examples of these roles are filling the background sonic canvas, generating repetitive bass with fast spatial movements, improvising within a constrained spectrum, improvisation through reaction towards the sonic gestures of musical AI, and staying quiet. Hence, the cues automatically set a structured improvisation setup in the complex performance environment of Revive. In the context of improvised (or nonidiomatic) music, structured improvisation is free improvisation with predefined constraints for musical sections. In addition, the synchronization of 3D audio spatialization with the audio-reactive generative visuals emphasizes the sonic gestures of performers in *Revive*. The specifics of 3D audio setup that we mention in this paper clarify the technical details of the performances at the Société des Arts Technologique (SAT) dome with 157 speakers clustered to 31 audio channels [2]. However, the spatialization tools in Revive is flexible for any 3D or 2D audio speaker setup.

In the following, we first give a brief introduction to the Creative Artificial Intelligence and Multi-agents Systems. We continue by explaining the performance setup of *Revive*. Then, we delve into the reactive agent architecture of the visual agents. We move further with the technical details and aesthetic background of sonic strategies in *Revive*. Finally, we conclude by the discussions around previously mentioned topics while proposing possible future steps.

2. MULTI-AGENT SYSTEMS IN CREATIVE ARTIFICIAL INTELLIGENCE FOR MUSIC AND MULTIMEDIA

Creative Artificial Intelligence (AI) for Music explores the applications of autonomous systems of Applied AI and Multi-agent Systems (MAS) for musical applications [2]. Autonomous system architectures for creative tasks differ from conventional architectures that aim to solve tasks with optimal solutions. Creative tasks often lack optimality, and the quality measures are ill-defined. For example, there is no universal objective measure to assess if one acousmatic composition is better than any other. The lack of objective measures asks for system designs where the possibilities of connections between autonomous behaviours and artistic aesthetics are explored. In many cases, the architectures work with a set of hyper-parameters where the user can manipulate the autonomous behaviour by exploring the space of these parameters.

Artifical agents in MAS are autonomous software with perception and action capabilities. Musical agents are implementations of Multi-agent systems combined with applied artificial intelligence and machine learning algorithms for musical applications. We previously clarified six levels of musical agent behaviours: 1- Reactivity, 2-Proactivity, 3- Interactivity, 4-Adaptability, 5-Versatility, 6-Volition and framing; where the higher levels can inherit properties of the lower levels [1].

The visual agent architecture in *Revive* aims for reactive behaviours whereas the musical AI, MASOM's initial architecture with VMM-PPM-C can exhibit interactive and adaptive behaviours. In Revive, we give up on the adaptive behaviours of MASOM with the Factor Oracle variant (MASOM-FO) to improve timbre consistency with mid-size datasets. The interactivity of MASOM-FO is more on a higher level through user interaction, where the user can change the statistical sequence model on the fly.

3. THE PERFORMANCE SETUP

Three sonic performers (including the musical AI MASOM) improvise in Revive and the performers are con-

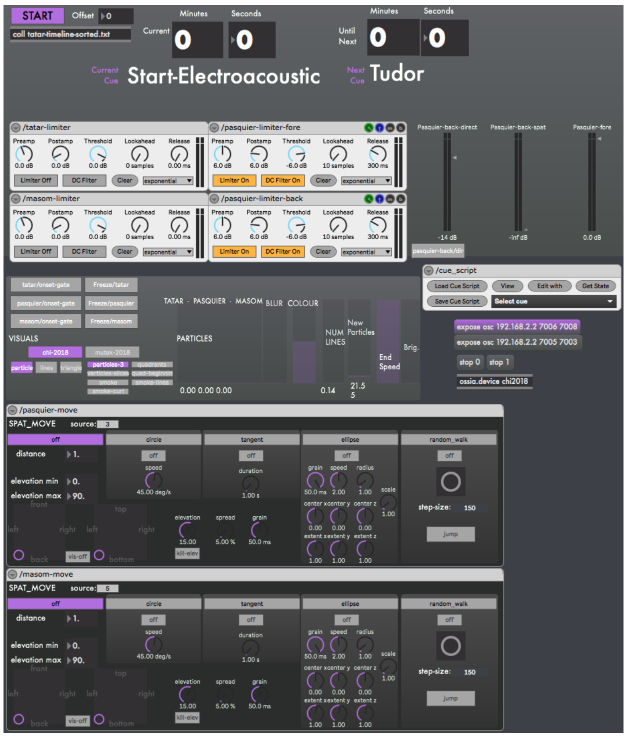


Figure 1: The UI of Revive's performance setup

strained within certain musical roles that are defined per musical sections. The performers apply various sonic strategies that we clarify in Section 5. The sonic actions of performers are visualized with a visual agent with audio-reactive behaviours. The localization of generated visuals follows the 3D spatial location of the audio. In addition to 3D audio with three sources, the second author also outputs a background channel that is directly send to all speakers. This helps to create a sonic background canvas for the performers in the foreground.

The Revive performance applies a cue system to handle parameter changes automatically (Figure 1). The cues also define musical sections where the sonic performers improvise within certain roles. The cues are automatically initiated at certain moments using a timeline. The cue system is implemented within the Jamoma [3] framework in Max [4] [3]. This framework automatically gathers all parameters of Max abstractions coded as Jamoma modules. The Jamoma's cue system allows a simple scripting where the parameters can be linearly ramped to any value. The script engine can also put into a halt for a certain time using the "WAIT" command. These features provide a simple, yet powerful coding of complex performance environments where many parameters constantly change. The cue system with Jamoma, sonic strategies in Revive, and machine listening modules of visual agents are implemented in Max. The visuals are generated and rendered to the dome view in Derivative's Touch Designer [5]. These two applications are networked through an OSC communication.

[2] http://sat.qc.ca/en/satosphere

[3] http://jamoma.org
[4] https://cycling74.com/products/max
[5] https://www.derivative.ca/

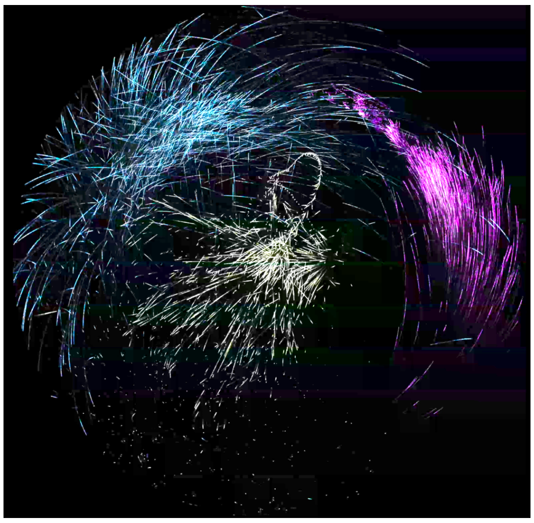
4. AUDIO-REACTIVE VISUAL AGENTS IN REVIVE

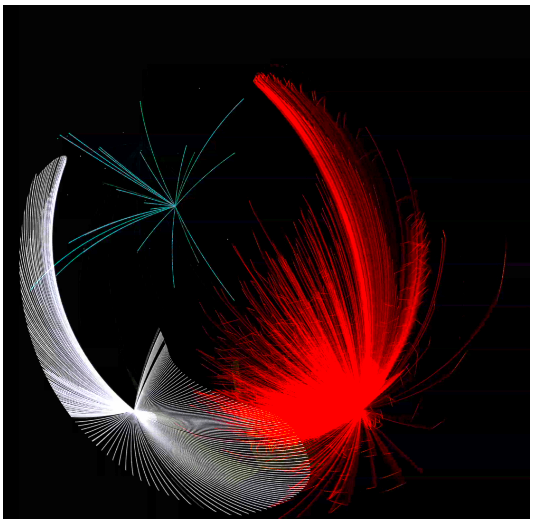
The visual agents aim to improve the audience's perception of sonic gestures. The visual agents are generative and reacts to the sonic performers by using a machine listening algorithm. Using reactive behaviours, three visual agents emphasize the actions of audio agents and make it easier for the audience to comprehend the connection between sonic gestures and the sonic performers' actions.

The visual agent architecture consists of a machine listening module for audio feature extraction and a particle engine to generate visuals. The location of particle engines' rendered outputs are synchronized with the spatial location of sonic performers in the 3D audio setup. As the sonic performers move in the 3D audio, the visuals follow the locations of sonic performers. During the Revive performances, three types of particle renderings provide variety in the visual content: 1- particles as sprites, 2- particles for drawing lines, 3- particles for drawing triangular shapes (Figure 2). The audio-reactive behaviours of three particle engines apply mappings of sonic performers' audio features to the input parameters of particle engines.

The machine listening modules of visual agents implement an onset detection with onset loudness detection and calculate continuous total loudness and specific loudness [4]. The onset detection uses the magnitude spectrum and if the summation of magnitude powers of all bands pass a user-set threshold, an onset is detected. Specific loudness is the loudness calculated for the Bark bands that is a 24-band spectrum that approximates critical bands of human hearing. We also apply a post-processing on the specific loudness to calculate a 3-band loudness spectrum of bass, middle, and high bands. The first two Bark bands are combined for the bass spectrum, the last eight bands are joined for the high spectrum, and the remaining fourteen bands constitute the middle band. The audio features are calculated for all sonic performers separately.

The audio features of a sonic performer are mapped to the parameters of corresponding particle engine. The total loudness is mapped to the overall particle speed and the brightness of particles. Two color schemes are applied: 1- static per performer, 2- the loudness of the mid-band controls the amount of green in red-green-blue color representation. Additionally, if the total loudness passes a certain threshold (above -3 dB for example), the type of noise that scatters the particles changes. This increases the overall movement of particles when the loudness moves closer to the maximum. Regarding rendering type 2 and 3 (Figure 2b and 2c), the particles have a life-span which is also a gaussian distribution. A static noise generates the birth locations of particles and the attraction points that the particles move towards within their life-time. The scale of birth locations is linearly mapped to the loudness of the bass spectrum. When there is more variance in the bass spectrum, the distances between the particle birth locations and the attraction points increase, which results in an increase in the fast movement behaviors of particles. The particles are re-initiated when the loudness of an onset passes a -6 dB threshold. As we mention in Section 3, the second author also plays a background channel that directly outputs to all speakers. Lastly, the total loudness of this channel


(a) Rendering type 1 - particles as sprites


(b) Rendering type 2 - particles for drawing lines

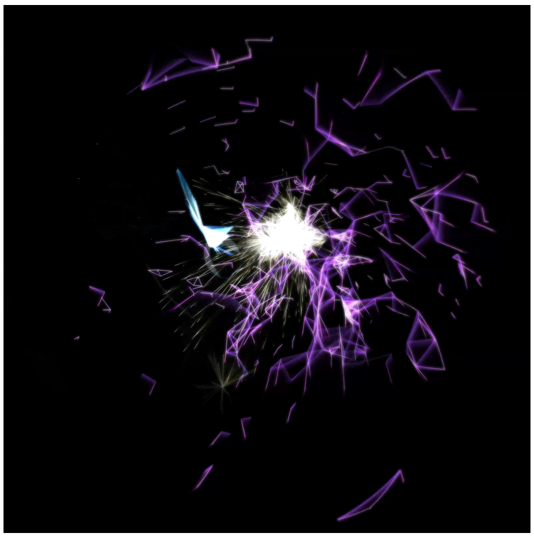

(c) Rendering type 3 - particles for drawing triangular shapes

Figure 2: Three snapshots of dome views illustrate three types of particle engine renderings. In each figure, three distinct colors corresponds to three visual agents that react to three sonic performers.

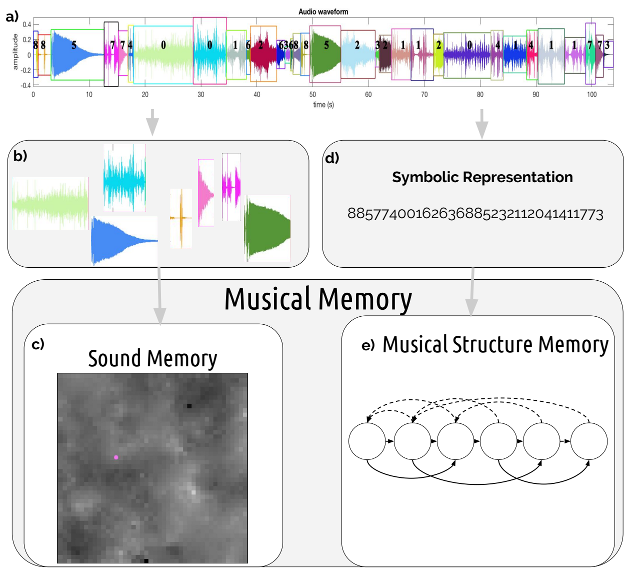


Figure 3: The offline learning in MASOM: a) Segmentation b) Labelling the audio samples c) Sound memory where squares stand for SOM nodes that are a clusters of audio samples d) Creating a symbolic representation of the original song using the clusters indexes of audio samples e) statistical sequence model to learn temporal transitions.

controls the amount of post-processing effects such as blur and feedback applied to all visuals.

5. SONIC STRATEGIES IN REVIVE

The first and the second author join MASOM in the sonic performance of Revive. These three performing agents apply different techniques and approaches, which come together through pre-defined roles within the structured improvisation. *Revive*'s aesthetics allow performers to improvise within these roles, and this introduces live sonic gestures back to the performance. In the following sections, we delve into the techniques of three sonic performers.

5.1 Musical Artificial Intelligence, MASOM-FO

Varèse defines music as "nothing but organized sounds" [5]. Inspired by this idea, MASOM's system design implements a neural networks algorithm combined with statistical sequence modelling algorithms in Max. The neural networks algorithm organizes the sound memory of the agent whereas the statistical sequence modelling algorithms handle the temporal musical structure modelling and user interaction. MASOM's unsupervised learning requires a set of audio recordings. The agent implements a Music Emotion Recognition algorithm in the machine listening module.

MASOM applies offline learning and online generation. The offline learning starts with segmentation of individual sounds in the recording (Figure 3a and 3b). The agent recognizes the audio segments between onsets as audio samples. MASOM use spectral magnitude based onset detection where an onset is detected when the summation of spectral magnitudes passes a user defined threshold. The implementation of segmentation and audio feature extraction uses IRCAM's MuBu Max Package [6] [6] and PiPo externals [7]. Following the segmentation, the training procedure labels audio samples with a 35-dimensional audio feature vector including:

- Timbre features: Perceptual Spectral Decrease and 13 Mel-frequency Coefficients (MFCCs)
- Fundamental Frequency
- Loudness
- Duration of audio sample
- Music Emotion Recognition (MER) features

Regarding timbre features, loudness, and fundamental frequency; we first calculate the features using a window size of 1024 samples and hop size of 256 samples, then we calculate the mean and standard deviation of these features per audio sample. The statistics of loudness, and 13 MFCCs, perceptual spectral decrease, and YIN-based fundamental frequency estimation [8] adds up to 32 features $((1 + 13 + 1 + 1) * 2 = 32)$. In addition, we add the duration of audio samples, and two MER features and the total number of audio features constitutes a 35-dimensional label vector.

The particular MER model that we use in the system design of MASOM is a two-dimensional, continuous multivariate linear regression model using the following equations:

$$\begin{aligned} Pleasantness = &-0.169+ \\ &-0.061 * Loudness_{mean} \\ &+0.588 * SpectralFlatness1_{mean} \\ &+0.302 * MFCC1_{std} \\ &+0.361 * MFCC5_{std} \\ &-0.229 * Percept.Spect.Decrease_{std} \end{aligned} \tag{1a}$$

$$\begin{aligned} Eventfulness = &-1.551 \\ &+0.060 * Loudness_{mean} \\ &+0.087 * Loudness_{std} \\ &+1.905 * PerceptualTristimulus2_{std} \\ &+0.698 * PerceptualTristimulus3_{mean} \\ &+0.560 * MFCC3_{std} \\ &-0.421 * MFCC5_{std} \\ &+1.164 * MFCC11_{std} \end{aligned} \tag{1b}$$

We generated this MER machine learning model using the Emo-soundscapes dataset that contains 600 curated soundscape recordings [8]. In the audio domain, the terms valence and arousal is exchanged with eventfulness and pleasantness, respectively [9]. This is mainly because of the contradiction that a sound does not feel an emotion, but stimulate an emotion. Hence, we can't label a sound with emotion categories of human cognition. For example, we can't talk about a happy sound (excluding anthropomorphism), but we can say that some sounds initiate happiness feelings in humans.

[6] `https://forumnet.ircam.fr/product/mubu-en/`
[7] `http://ismm.ircam.fr/pipo/`
[8] Please refer to [4] for the details of audio feature calculations and [7] for the YIN algorithm.

Following the segmentation and labelling audio samples, the agent trains a Self-Organizing Map to create a latent space of sonic possibilities (Figure 3c). In this latent space, similar sounds locate close to each other. Self-Organizing Maps are fully connected artificial neural networks with unsupervised learning[10, 11]. SOM is used for visualisation, representation, and clustering of high-dimensional input data with a 2D topology of square, rectangular, toroid, and arbitrary shapes (such as Mnemonic SOMs). SOMs consist of a number of nodes that position themselves during training to represent the topology of the input data. The nodes are vectors with the same number of dimensions with the input data. Hence, SOM creates a symbolic latent space that represent the topology of the training data.

MASOM incorporates the SOM implementation included in *ml.star* Max Package [12]. Regarding the SOM training [9], we first normalize the input data using the equation 2:

$$I_{norm}[i] = \frac{I[i]}{M[i]} * STD[i] \tag{2}$$

and $i \in [1 : N]$, where the N is the total number of dimensions of the input vector, I is the calculated audio feature vector, M is a vector that gives the average of each audio feature, STD is a vector of standard deviation of each audio feature, and I_{norm} is the normalized feature vector. M and STD are calculated for a given corpus.

Then, we apply a weight vector on the normalized input data of SOM. In the weight vector W, MFCC features are multiplied by $1/13$ so that combined MFCC distances affect the SOM training as one timbre feature. The rest of the features are kept at the original value.

$$a = int(\sqrt{\text{the number of audio samples in the memory}/6}) \tag{3}$$

After the the pre-processing, we train an SOM map with square topology, $a * a$ where a is found by using the equation 3 that aims for 6 samples per cluster. We found that this approach consistently gave a low number of SOM nodes where no audio sample was clustered. The total number of epochs in the SOM training is 1000. The learning rate and the neighbourhood drops from 0.25 to 0.01 and $a/4$ to 0, respectively.

Clustering follows the training of SOM sound memory. We calculate Euclidean distance between SOM node vectors and thumbnail vectors of audio samples in the agent's memory. The audio samples are labelled with the SOM node that gives the lowest Euclidean distance. The numbers within the segment squares in Figure 3a are the closest SOM node indexes. Following this labelling, we generate a symbolic representation of the original song using the sequence of SOM node indexes (Figure 3d). This sequence is later used to train the statistical sequence modelling algorithm.

The agent can generate compositions on the fly and change musical structure models through user interaction. A statistical sequence modelling algorithm learns and generates SOM node index sequences (Figure 3d). These nodes are clusters of audio samples, and the agent chooses a sample clustered by an SOM node randomly. MASOM's training aims for 6 audio samples per node, and the sonic variation within a node is constrained because of the unsupervised learning. Hence, the random sample selection within the SOM node audio cluster implements a type of constrained sonic variation in the agent's output.

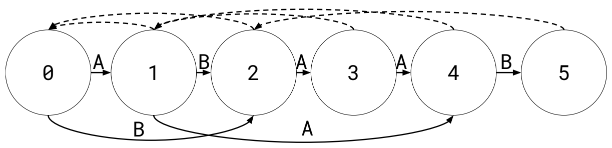


Figure 4: Factor Oracle generated using the sequence $ABAAB$.

MASOM's architecture was initially using Variable Markov Models (VMM) Prediction by Partial Matching C (PPM-C) variant [13]. MASOM VMM PPM-C variant was trained on small and medium size audio corpora ranging from an album to approximately 1GB of audio recordings (lossless stereo wave files). Although the interactive nature of MASOM VMM PPM-C variant was satisfying enough for public concerts with small and medium size corpora, we realized that this variant lost timbre consistency with big size corpora ranging from 1GB to 100GB of audio recordings. Hence, we switched MASOM's statistical sequence model to a generative Factor Oracle algorithm to ensure timbre consistency and we refer to this variant as MASOM-FO.

Factor Oracle, initially proposed as a compression algorithm, is a statistical sequence modelling algorithm as well as a finite state automata [14]. FO models repeating patterns in a sequence, which are the *factors* of a sequence. FO has three types of links: internal links (forward links between successive states), external links (forward links that jumps to a future state), and suffix links (backward links, dashed lines in Figure 4). Suffix links marks the longest repeating factor in previous states. FO allows incremental learning, and learning is linear in time and space [15]. Several musical agents previously implemented FOs in the architecture [1, p. 26-29].

The training of FO allows a single sequence for training. In MASOM's case, the agent's corpora include several audio recordings and the symbolic SOM node representations of these recordings [10]. This constraint emerges two options for FO training: 1- concatenating several (or all) symbolic representations of audio recordings to one sequence, 2- using the symbolic representation of only one recording. The first option risks timbre consistency as various audio recordings cover a wide range of timbre possibilities, even within the same musical style. Hence, we apply option two in Revive project to ensure timbre consistency.

MASOM-FO incorporates Wilson's [16] Max external implementation which is sufficiently fast for real-time training. In Revive's structured improvisation, MASOM's role in the performance is constrained by the FO training. In each section, we train the agent's FO from scratch using the symbolic representation (the sequence of sound cluster indexes) of one audio recording. This results in a high-level user interaction where the user defines the musical constrains and roles of the agent by forcing a subset of sound clusters, and the temporal patterns and structures of an audio recording. The subset of sound clusters may contain samples of other recordings due to the SOM training,

[9] The details of SOM training procedures is available at [11, 13].

[10] The details of FO training is available in [16].

and this introduces variety in the agents audio output.

FO initiate SOM node (cluster) index generation with sound selection using two approaches: 1- playing one sample after another, 2- user-defined time intervals. The first approach outputs one layer of sound events where each sample is concatenated one after another. This approach generates a monophonic output as the FO waits for the previous sample to finish before initiating the next one. The second approach creates a multi-layered output where the user can change the audio event density by manipulating the time intervals between sample initiations. The agents uses FO to output a sound cluster index in user-defined intervals. Using any of these two approaches, FO outputs a sound cluster index. MASOM chooses a sample within that cluster randomly. The random picking of samples is a constrained selection because of the SOM clustering, and it aims for the generation of constrained variation in the agent's output. In Revive, these two generative approaches are initiated by performance cues. We carry out the audio event density manipulation in the second approach by applying linear ramps on the time interval parameter.

5.2 Distruption of fixed-media

Acousmatic music facilitates electronic means to create or process sounds to produce musical compositions. The public presentations of acousmatic music happens as playback of the compositions without any interaction, or live mixing of prepared tracks using volume adjustments, fade ins and outs, equalizers, and spatialization [17]. The second author puts a rich MAX-based sampler player, Kenaxis [11] into practice with samples of analog machines such as EMS Synthi AKS, Korg Mono/Poly, Serge and Eurorack analog modular synthesizers, as well as field recordings of forests, waterfalls, and thunderstorms. Samplers and granular synthesis engines are controlled live with a BCF-2000 MIDI controller. The setup allows a sonic palette of background layers made of drony textures, as well as foreground gestures including more melodic motifs with live effects.

The reflections of the third wave of HCI studies have initiated the introduction of embodied interaction for live music performances [18] and proposed a combined understanding of sonic and bodily interaction [19]. In line with the live performance trends in computer music, the first author's live sonic performance transforms his fixed-media compositions to sonic material for improvisation combined with live generation of sonic gestures. In this self-disruption process, the first author metamorphoses his previous fixed media pieces to a sonic vocabulary for live improvisation of experimental electronic music. To do so, the first author combines wavetable synthesis with a game controller interface to improvise experimental electronic music on a 3D speaker configuration (Figure 5).

Wavetable synthesis uses varying playback speeds to manipulate an audio buffer to synthesize sounds. Varying playback speed generates pitch-shifting and time-stretching audio manipulations, and it is possible decorrelate these two manipulations [20]. The wavetable synthesis utilize an audio buffer and allows sudden changes of the audio in the buffer. The first author's performance practice in Revive applies one-minute long excepts from his previous fixed media compositions (Figure 5b). The audio buffer of wavetable synthesis is a certain portion (window) of these one minute excerpts. This approach provides a sonic diversity using varying window sizes and positions using a game controller as the interface.

The first author utilizes an XBox controller that includes two joysticks buttons underneath, one d-pad, 9 additional buttons (*x,y,a,b*, left-button, and right-button), and two triggers that acts as sliders, illustrated in Figure 5a. The *joystick-2* handles the interaction with the spatialization and the rest manipulates the wavetable synthesis. Although the *joystick-2* outputs a 2-dimensional spatial position, we project 2D positions on a 3D spherical surface to create 3D spatialization trajectories in azimuth-elevation-distance format using the equations,

$$r_{2D} = \sqrt{(position\_x_{2D})^2 + (position\_y_{2D})^2} \tag{4a}$$

$$azimuth_{3D} = arccos(\frac{position\_x_{2D}}{r_{2D}}) \tag{4b}$$

$$elevation_{3D} = \frac{\pi}{2} * (1 - r_{2D}) \tag{4c}$$

where r_{2D} is the distance from the center in 2D, $azimuth_{3D}$ and $elevation_{3D}$ ranges are $[-\pi, \pi]$ and $[0, \frac{\pi}{2}]$; respectively. The distance is static, and this ensure 3D spatialization while preserving the original loudness of the audio source.

The y-axis of *joystick-1* controls the amount of pitch-shifting, and the x-axis controls the multiplier of time-stretching. When *joystick-1* is in the resting position, the wavetable synthesis plays the original content in the audio buffer. The button on the *joystick-1* plays the sound; hence, there is no sound coming out when the user moves the hands away from the interface. The button *x* reverses the audio buffer. The button *b* allows a toggle for continuous audio play for occasion where a sustained continuous playback is needed, such as textural sounds that functions in the sonic background. Hence, the performer can explore a continuum of sonic choices from gestural foreground actions to textural background material. The up and down buttons on the *d-pad* adjust the output volume, and left and right buttons change the source spread in the 3D spatialization. The spread controls the total area of audio source diffusion. The buttons *a* and *y* set the spread to 1% and 90% in 2-seconds, respectively. The performer can cycle through several audio buffers using *start* and *back* button. Lastly, the *home* button resets the output volume to 0 dB in 3-seconds.

5.3 3D audio spatialization techniques in Revive

Sonic performers in Revive are spatializaed in 3D speaker setups using IRCAM-SPAT Max library [12] [21]. This library is fast and flexible to change 3D audio speaker setups and test spatialization methods in soundchecks and rehearsals where the time is limited. In Revive, we use 3D vector-based amplitude panning (vbap3D). The audio source positions are processed using the azimuth-elevation-distance format. The distance is static so that performers change the loudness of their output as they prefer. Throughout this paper, the elevation range is $[0, \frac{\pi}{2}]$ that addresses

[11] `https://www.kenaxis.com/products/kenaxis-2/`

[12] `https://forumnet.ircam.fr/product/spat-en/`

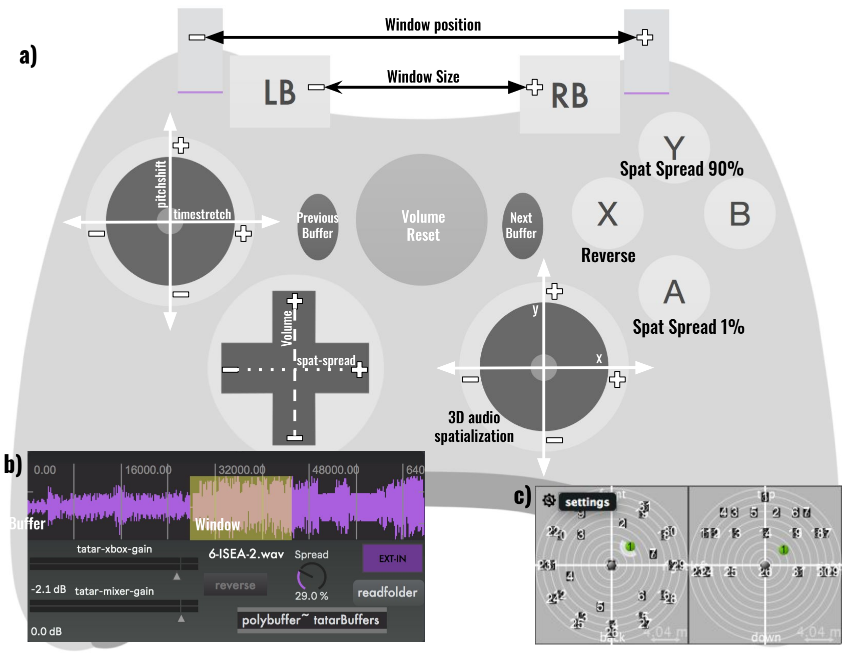


Figure 5: The framework for using fixed-media compositions as content for the live audio-visuals with 3D audio: a) the game controller interface and the mappings b) wavetable synthesis c) 3D audio visualization with the SAT dome setup.

the setup of the SAT dome. The elevation range is an input variable for all spatialization modules for different 3D speaker setups.

The first author controls the spatial location using the joystick on the game controller. The output of second author and MASOM are positioned using three different generative trajectories: circular, tangential, and random-walking. The movement speed is a variable in all three cases, and it is changed with the cue system of Revive. The first method cycles the azimuth using a speed variable (degree/s) and the elevation can be fixed or adjusted by the cue system. The second approach, tangential trajectory generation chooses a random location on the opposite half of the sphere in relative to the current source position, and the source moves to this location in a duration that is set by the cue system. The azimuth (azi) and elevation (el) of the new location is generated using the following formula:

$$azi_{new} = random(azi_{current} + \frac{\pi}{2}, azi_{current} + \frac{3\pi}{2}) \qquad (5a)$$

$$el_{new} = (el_{current} + random(0, \frac{\pi}{4})) \mod \frac{\pi}{2} \qquad (5b)$$

where the function $random(a, b)$ generates pseudo-random values in the range of $[a, b]$. The third approach, random-walking first generates a step size within a user-set range using the pseudo-random function. The generated step size is added to the current location of the source; and thus, the source jumps to a new location. The steps in random-walking can be triggered by two ways: using a fixed duration or using a magnitude spectrum based onset-detection. In case of MASOM, we apply an additional approach where the random-walking is triggered with every sample initiation in MASOM. The trajectory generation selections and their input variables are controlled by the cue system during the performance.

6. CONCLUSION

We introduced the live audio-visual performance project *Revive*. The medium of this audio-visual performance project is a dome projection with 3D audio setup where the audio and the visuals are synchronized in position. *Revive* fuses a musical AI architecture into structured improvisation for audio-visuals. The musical AI in Revive, MASOM employs Factor Oracle algorithm for temporal content generation and user interaction. Thomas et al. [22] compare Factor Oracle, Fixed-Length Markov Model, and MusiCOG on melody generation and show that these sequence modeling algorithms introduces particular biases to the sequence generation. In our future work, we plan to compare various statistical sequence modeling algorithms in terms of sub-sequence cloning, that is how much the model copies the patterns in the training dataset.

The live performance of Revive benefits from an automatized parameter management using Jamoma's cue system. The compositional approaches of *Revive* allows performers to improvise within predefined roles. During the development of *Revive*, the task with the highest time-complexity was the setup and exploration of the mapping between audio features and input parameters of generative visuals. Although several mapping tools have been developed previously, these tools are not necessarily made for exploration and comparison of several mapping possibilities. To address this issue, several researchers around the globe planing to collaborate with the OSSIA [13] initiative. OSSIA, stands for Open Software System for Interactive Applications, is an OSC-query based open-source framework for time-scripting and mapping for interactive scenarios. The framework is currently in its alpha stage, and we hope that the project will move further in future.

[13] `https://ossia.io/`

Acknowledgments

This research was funded by the Natural Sciences and Engineering Research Council of Canada, Social Sciences and Humanities Research Council of Canada, Canada Council for the Arts, and Swiss National Science Foundation.

7. REFERENCES

[1] K. Tatar and P. Pasquier, "Musical agents: A typology and state of the art towards Musical Metacreation," *Journal of New Music Research*, vol. 48, no. 1, pp. 1–50, Sep. 2018.

[2] P. Pasquier, A. Eigenfeldt, O. Bown, and S. Dubnov, "An Introduction to Musical Metacreation," *Computers in Entertainment*, vol. 14, no. 2, pp. 1–14, Jan. 2017.

[3] T. Lossius, T. de la Hogue, P. Baltazar, T. Place, N. Wolek, and J. Rabin, "Model-View-Controller separation in Max using Jamoma," in *Proceedings of the joint 40th International Computer Music Conference & 11th Sound and Music Computing Conference,*, Athens, Greece, 2014, p. 8.

[4] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the CUIDADO project," IRCAM, Tech. Rep., 2004.

[5] E. Varèse and C. Wen-chung, "The liberation of Sound," *Perspectives of New Music*, vol. 5, no. 1, pp. 11–19, 1966.

[6] N. Schnell, A. Rbel, D. Schwarz, G. Peeters, and R. Borghesi, "MuBu and friends-assembling tools for content based real-time interactive audio processing in Max/MSP," in *Proceedings of International Computer Music Conference (ICMC)*, 2009.

[7] A. de Cheveigné and H. Kawahara, "YIN, a fundamental frequency estimator for speech and music," *The Journal of the Acoustical Society of America*, vol. 111, no. 4, p. 1917, 2002.

[8] J. Fan, M. Thorogood, and P. Pasquier, "Emo-soundscapes: A dataset for soundscape emotion recognition," in *2017 Seventh International Conference on Affective Computing and Intelligent Interaction (ACII)*. San Antonio, TX: IEEE, Oct. 2017, pp. 196–201.

[9] ——, "Automatic Soundscape Affect Recognition Using A Dimensional Approach," *Journal of the Audio Engineering Society*, vol. 64, no. 9, pp. 646–653, Sep. 2016.

[10] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biological cybernetics*, vol. 43, no. 1, pp. 59–69, 1982.

[11] ——, "The self-organizing map," *Neurocomputing*, vol. 21, no. 13, pp. 1–6, Nov. 1998.

[12] B. D. Smith and G. E. Garnett, "Unsupervised Play: Machine Learning Toolkit for Max." in *the Proceedings of International Conference on New Interfaces for Musical Expression 2012*, 2012.

[13] K. Tatar and P. Pasquier, "MASOM: A Musical Agent Architecture based on Self Organizing Maps, Affective Computing, and Variable Markov Models," in *Proceedings of the 5th International Workshop on Musical Metacreation (MUME 2017)*, Atlanta, Georgia, USA, Jun. 2017.

[14] C. Allauzen, M. Crochemore, and M. Raffinot, "Factor oracle: A new structure for pattern matching," in *International Conference on Current Trends in Theory and Practice of Computer Science*. Springer, 1999, pp. 295–310. [Online]. Available: http://link.springer.com/chapter/10.1007/3-540-47849-3_18

[15] A. Lefebvre and T. Lecroq, "A Heuristic For Computing Repeats With A Factor Oracle: Application To Biological Sequences," *International Journal of Computer Mathematics*, vol. 79, no. 12, pp. 1303–1315, Jan. 2002.

[16] A. J. Wilson, "factorOracle: an Extensible Max External for Investigating Applications of the Factor Oracle Automaton in Real-Time Music Improvisation," in *Proceedings of the International Workshop on Musical Metacreation 2016*, Paris, France, 2016, p. 5.

[17] E. R. Miranda and M. Wanderley, "Toward Intelligent Musical Instruments," in *New Digital Musical Instruments: Control And Interaction Beyond the Keyboard*, 1st ed. Middleton, Wis: A-R Editions, Inc., Jul. 2006, pp. 219–255.

[18] P. Dourish, *Where the Action Is: The Foundations of Embodied Interaction*, 1st ed. Cambridge, Mass.: The MIT Press, Aug. 2004.

[19] A. R. Jensenius and M. J. Lyons, Eds., *A NIME Reader: Fifteen Years of New Interfaces for Musical Expression*, ser. Current Research in Systematic Musicology. Springer International Publishing, 2017.

[20] "zynaptiq: ZTX Features And Specifications." [Online]. Available: https://www.zynaptiq.com/ztx/ztx-features-and-specifications/

[21] T. Carpentier, M. Noisternig, and O. Warusfel, "Twenty Years of Ircam Spat: Looking Back, Looking Forward," in *Proceedings of the 41st International Computer Music Conference (ICMC 2015)*, Dentan, Texas, USA, 2015, pp. 270–277.

[22] N. G. Thomas, P. Pasquier, A. Eigenfeldt, and J. B. Maxwell, "A Methodology for the Comparison of Melodic Generation Models Using Meta-Melo." in *Proceedings of the 14th International Society for Music Information Retrieval Conference*, Brazil, 2013, pp. 561–566.