Cocreative Interaction: Somax2 and the REACH Project

Gérard Assayag, Laurent Bonnasse-Gahot, Joakim Borg

Somax2 : REACHing Cocreative Interaction

Gérard Assayag (1), Laurent Bonnasse-Gahot (2), Joakim Borg (3)

(1) (3) IRCAM - STMS, (2) EHESS - CAMS

Introduction

The REACH (Raising co-creativity in cyber-human musicianship) project aims at modeling and enhancing co-creativity as it arises in improvised interactions between human and artificial agents (or between agents), through a wide spectrum of practices spanning from performing live with computers to mixed reality involving instrumental physicality and musician embodiment. REACH is a wide ranging project funded by the European Research Council, directed by Gérard Assayag and led in partnership with UCSD (Pr Shlomo Dubnov) and EHESS (Pr Marc Chemillier).

Modeling and enacting improvisation through AI algorithms (Assayag 2021, Dubnov 2021, Chemillier 2021) has been an ongoing activity at Ircam in the group led by Assayag for a number of years, with a conception of human-machine co-improvisation founded on structuring oppositions such as : anticipation / surprise, reactivity / planification, discovery / action, purpose / conformism etc. These qualities inspired by observation of great human improvisers have raised significant research challenges when it came to master them through engineered, cognitive or machine learning approaches.

These works have led to a family of operational systems such as the Omax environment (Assayag & al 2004, 2006) and its many heirs (Nika & al 2017) that have made their mark as evidenced by the number of scientific and artistic production (more than a hundred public performances including world-class — human — musicians and concert places).

From there, REACH is now pushing further the boundaries in three main

directions : integration of deep learning methods (VAE, transformers, Gan, etc.) to deepen the latent spaces underlying these systems' musical knowledge (Dubnov & al 2022, Dong & al 2023, Carsault & al 2021) ; developing the self-reliance (in the sense of agent autonomy) of artificial players to enhance their co-creative offerings in interaction ; and set a strong social sciences research context to observe, interpret, and get feed-back from the new praxis induced by these radically new directions in music (Chemillier 2021).

Achievements of REACH in the software domain so far are the recent finalization of Dyci2 (Nika & al 2017) as a full scale environment and the inception of Somax2. This article will focus on Somax2.

After exposing the research methodology of REACH, from the epistemological underpinnings to the learning strategies, the article will proceed shortly through the genealogy stemming from the founding Omax interaction paradigm, explaining the specificity of each new branch that split up from it in terms of diversification of cognitive improvisational abilities, then the Somax2 philosophy, model and architecture will be described.

REACH

As for the working hypothesis, REACH is based on the idea that co-creativity in cyber-human systems (Assayag 2020, 2021, Lewis 2021) results merely from the spontaneous emergence of joint behaviors in collective settings, building up non-linear regimes of audible structural occurences, leading to rich musical co-evolution of forms. We assume that this distributed creation is induced by cross-learning mechanisms between (artificial or human) agents, involving multiple feedback loops and reinforcement channels. As is usually the case for emergence phenomena in complex systems, the resulting musical dynamics is not reducible to, nor explainable by the mere observation of individual agents' behavior : it is a transitional, sometimes ephemeral effect of the interaction itself and as such

intrinsically a co-creative production. This recalls of course Norbert Wiener's Cybernetics as self-guided systems, with an emphasis on the idea of cross-feed-backs and influence between agents (Wiener 1948).

By producing emergent information structures as a result of cyber-human interaction, we might achieve an epistemological leap beyond the classical philosophical difficulty of conceding the faculty of creativity to artificial systems (which are not reflexive subjects — yet), by assessing that creativity is not a state anyway, but rather a highly dynamical (and ephemeral) effect of agent interaction in a complex system. This is why we put forward the term "co-creativity", enforcing the idea (often noted in computer music) that a highly creative system can very well be based on interacting components that are pretty simplistic themselves.

We put in place a multi-disciplinary (music, computer science, social sciences, cognition) research ecosystem allowing us to answer two central questions:

- How to augment the digital agents capacities with enhanced computational creativity and cyber-physical extensions, so they can develop convincing interactions with humans ;
- How to augment the human capacities by expanding their individual and social creative potential through novel collaborative strategies and mixed realities, so they can naturally immerse themselves in complex settings involving digital intelligence.

In effect, we believe that "computational creativity" (Wiggins 2006) will socially take root when it eventually creates the conditions of emergence of cyber-human co-creativity as presented above, and when it actually clings to an augmented physical reality allowing users to experience a rewarding embodied relation.

Somax2 is an early step in that direction allowing rich improvised interaction scenarios that we can experiment in real-life performances with top-level musicians bringing invaluable feed-back and musical ideas. Somax2 is quickly gaining

momentum in professional scenes with major artists such as Bernard Lubat or Joëlle Léandre (whose lifetime achievement award ceremony is organized in 2023 in New York) actively experimenting and creating content, with concerts scheduled at mainstream venues such as IRCAM - Centre Pompidou in Paris and the Improtech Festival where the software will be on stage with the legendary Evan Parker in addition to the above. Other scenes where the software has been performing at different stages of development include the Onassis Center in Athens, the Annenberg Center in Philadelphia, the KLANG festival in Copenhagen, the Philarmonie de Paris Concert Hall.

Audio and video examples of recent artistic experiments with Somax2 as well as Somax2 installation, sources (Somax2 is OpenSource) and demos can be found here:

https://www.stms-lab.fr/article/somax2-examples-for-cmj

Similar Works

The idea of improvising agents fed by AI-inspired algorithms and likely to enable situations of musician machine co-creativity has been pioneered by George Lewis with his system Voyager in the eighties (Lewis 2000, 2021). Lewis' agents were inspired by AI developments of the time and reproduce separate cognitive functions that cooperate or compete to make musical decisions. Since then several improvisation environments have been proposed. Tatar and Pasquier have published a general survey on "Computational Creativity, Multi-Agent Systems and Artificial Intelligence" (Tatar & Pasquier 2018). They have coined (along with Shlomo Dubnov, part of our REACH project) the expression "Musical MetaCreation" for this research field, which significantly intersects the topics discussed in this article. These authors have themselves used in some of their works the SOM Model discussed further in the Somax2 presentation (Tatar & Pasquier 2017). Following the agent classification they propose, the Somax2 system would be considered

ultimately hybrid since it combines architectural traits usually found separately : statistical learning (Conklin 2003), cognitive algorithm (Maxwell & al 2012), neural/SOM representation space (Briot 2019) and reflexivity through self-listening (a trait we have not seen elsewhere). Its interaction strategies encompass as well the three different modes identified by the authors : learning from humans, controlled by humans, and playing with humans — that may strongly contribute to a lively improvisation environment (Blackwell & al 2012). Somax2 pertains to the REACH co-creativity paradigm, where creativity can be understood only through the emergent properties distributed agency in live interaction, a depart from classical studies on artificial creativity (Wiggins 2006). As such Somax2 is to the best of our knowledge a unique and rich system, complete with a novel model of dynamic musical memory and advanced coordination of listening, recognition, matching, anticipation, and decision skills.

The Omax galaxy

Somax2 belongs to a family of related research and implementations on machine improvisation. They share a sequential model, that we call “memory”, learned from live or offline music streams, that is explored interactively at performance time with various types and degrees of constraints. These system stem from the seminal Omax system (Assayag & al 2006).

Omax was dedicated to non-idiomatic and non-pulsed improvisation. It learns typical features of a musician's style in real-time (or off-line) and plays along with them interactively, giving the flavor of a machine co-improvisation. Omax uses a fast statistical sequence model delivering a semantic representation and allows immediate recombination and transformation of the captured material. We coined the term Stylistic Reinjection to express this particular way of interacting with one's musical clone. Omax listens to learn, but does not listen to play, and it plays stochastically the model learned, so it is "low reactive" and "low planning". A first

descendant of Omax was ImproteK (Nika, Chemillier, Assayag 2017) now known as Djazz. In addition to Omax's listening and modeling capacities, Djazz features the notion of long term scenario and beat structure, where a scenario is an external sequence of labels in the same symbolic language as the representation learned. Djazz creates anticipations in the model that best match both the future of the scenario and the memory structure, and reconsiders them depending on the evolution of the context. DJazz is not reactive and thus "low reactive" "high Planning".

Dyci2 was intended in the first place as a way to synthesize all these different flavors of an improvising musician ("high reactive" "high planning" while keeping all the fluidity of Omax free improvisation skills), in an integrated approach, but it actually evolved to yet a new improvisation paradigm, introducing the notion of short-term scenario ! Dyci2 builds on general audio-descriptors clustering methods and lets the user specify micro-scenario, expressed in this descriptors classes language, that are triggered as response to input flow. It is moderate on reactivity and on scenario.

Thus Somax2 is the "high reactive" "low planning" environment described in this paper as we will see in the next section. It implements a complex memory model allowing to identify and evolve through time a collection of matches between the input and the model, and to provide both fast response and a sense of anticipation that make it fit for improvisation.

Before all these interaction paradigms get a chance to be integrated in a full scale co-improvisor, we have pushed every of them to become highly sophisticated specialized environments, and it is not uncommon that we use several of these programs simultaneously in concert in order to cover the different flavors of improvisation. Figure 1 shows the Omax paradigm legacy and the large amount of research and development that it has inspired in many places.

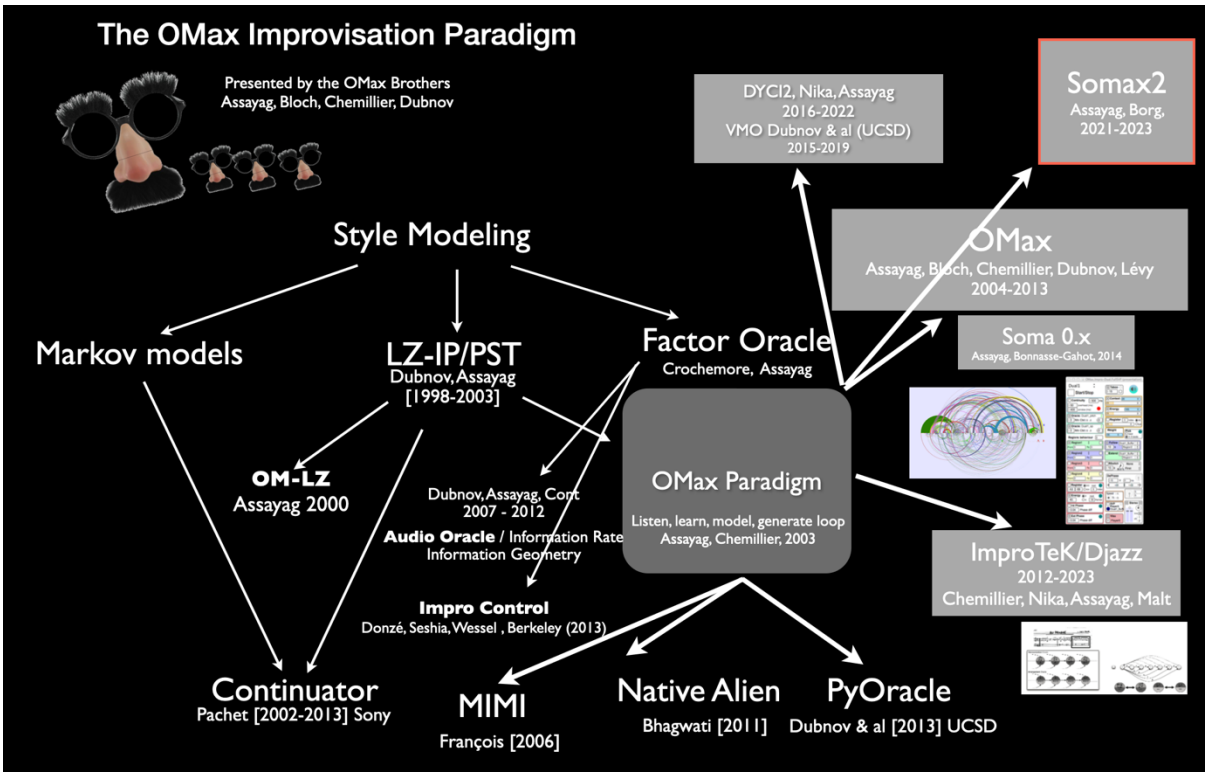

Figure 1. The Omax legacy.

Somax2

As main characteristics with regard to its siblings, Somax2 features hyper-reactive AI agents that may listen to, make musical sense from, and react to human musicians or other likewise agents (including themselves as a source of self-assertiveness. By this we mean that there's a "self-listening" channel creating inferences that can contradict the external influences, just as it could happen with humans !). This strong autonomy is balanced by their instrumentality, that is the capacity for humans to take control and continuously drive their generative power as they would do for an instrument, so there's a variety of flavors in Somax2 from pure generativity to instrumented live electronics action. As for it siblings, Somax2 is a memory based system, that is it creates new content by forcing a new way through material it has learned off-line or live, based on statistical sequence

modeling (Assayag 2004, Déguernel & al 2018, 2019) and/or music information dynamics (Wang & al 2016, Dubnov 2021, 2022).

Somax2 has been provided with a set of decisive features structured around 5 mains "skills" :

1. a latent space built once for all by machine learning algorithms trained on large musical data collections, encoding the general harmonic and textural knowledge ;
2. a discrete sequential learning model able to figure out the pattern organization of musical streams and form a state-based memory structure (a local or stylistic knowledge);
3. a dynamic cognitive memory model able to evolve continuous activation maps over the state structure, representing the hotspots reacting to ever shifting internal and external influences and viewpoints ;
4. a real-time machine listening device able to segment, analyze and encode musical streams in discrete components continuously matched against the memory through the latent space;
5. a set of interaction policies determining how and when to react to influences.

Creative agents (called Players) in Somax2 combine these skills in order to acquire stable semantic knowledge, create perceptual communication channels with the external world or between themselves, perceive or generate musical influences that will condition their individual reactions to an evolving context and to the fast transformations of their internal states. The AI learning aspects of these skills and the technical architecture by which agents combine them into highly efficient information flows and distributed processing will be described below.

As for skill 1 (training), a collection of copyright-free Midi files with solo and ensemble music from baroque to XXth century was used in the first place. Although

this obviously involves cultural bias toward western tradition, this impacts only the textural / harmonic features used for matching. Continuous pitch tracking and the possibility to load agents with audio materials from any tradition makes Somax2 fit to co-improvise in other settings, as has been experimented already with Maqâm or Indian-Japanese instrument ShahiBaaja.

As a reactive environment, Somax2 uses at its core the concept of "influence" understood as an interpretation of the raw musical content (audio, midi) circulating in the system, through multiple viewpoints chosen from available musical dimensions (melody, harmony, texture, timbre, rhythm, etc.). Flows of influences coming from multiple sources — including the "self" — constantly reshape the agents' cognitive state and determine their purpose and their responsiveness in the context of a global performance where a collectivity of human and artificial subjects participate and bond together. This influence system acts asynchronously on the memory model by continusously shaping the activation profile, and all sources of influence (musicians, other agents, self-influence) are processed in a concurrent way and merged into the current profile. The decision to play at time *t* is conditioned by this rapidly evolving memory map mixing a great number of factors into a probability distribution. This influence system that somehow mimics the high parallelism of the brain is unique to Somax2 to the best of our knowledge.

It proves to be a powerful way to evolve co-creative scenarios, as documented in audio and video archives we are preparing to make available to social science and experimental studies, such as the one we carried on improvisers concordance strategies in (Golvet & al 2021). It is, in effect, fascinating to see how highly skilled human performers reshape their musicianship when interacting with AI algorithms trained on familiar or uncommon musical content (Bloch & al 2018).

Somax2 Multi-Learning Model

Somax2's learning model is twofold. A general latent space embeds musical knowledge learned once and for all with a model trained on large collections of pieces spanning five centuries of western music. Agents are specialized by learning specific materials that stylistically inform them and give them personality. So an agent always keeps a course between the general knowledge (of a textural nature), a specific style (navigating a musical memory subspace), and a live context (constraining the navigation by listening and reacting).

AS for the general textural knowledge, a SOM (self organising map) and a Convolutional VAE (variational autoencoder) have been experimented with so far.

SOM's (Kohonen 2013) are pre-deep learning, unsupervised neural networks using competitive learning and producing topologically organized low dimensional feature spaces. They builds on biological models of neural organisation resembling cortical feature maps (Bednar & Wilson 2016). Toivianen pioneered the musical use of SOM's (Toivianen 2005) to correctly map tonal centers, using a toroidal topology where close cells mapped close music-theoretic keys with fifth, relative, or parallel relations. We used a more massive approach, feeding the network with hundreds of midi-files segmented in beat units reduced to chroma vectors computed from the pitch content enriched by pitch harmonics up to a certain rank. As a result, 3600 chromas were clustered into 121 SOM "textural" classes. While keeping the underlying tonal reference as a baseline, the idea was to provide a textural mapping engine that would apply to complex polyphonic or timbral mixtures as well in the symbolic (Midi) or the audio domain (where constant-Q chromagrams are used).

As for VAEs (Diederick 2019) an autoencoder is a non-supervised machine learning model consisting in an encoder e which compresses the data x in a lower-dimensional (latent) space, and a decoder d that "reconstructs" the data such that $\hat{x} = d(e(x)) \approx x$. A variational autoencoder (VAE) consists in an inference (encoder) and a generative (decoder) deep neural networks that parametrize the likelihood

and approximate the posterior of a latent Gaussian model. Since latent codes are parameters of probability distributions (mean and deviation) for each data sample, VAEs latent space naturally underpins a Riemannian manifold structure (where points are probability densities) susceptible to be studied by information geometry (Chadebec & al 2022). So a latent code is really a distribution from which a new sample can be generated.

Using a corpus of J.S. Bach Chorales available in Music21, and reducing the corpus again to beats-chromas (enriched by harmonics) "slices", we trained a β-VAE with 2D convolutional layers, effectively shaping a structured latent space of chromas (Feldman 2021). In order to evaluate this space and compare it to the SOM, we built a labeled synthetic data set of chromas, comprising of the 24 major and minor chords enriched harmonically, and projected this set into the VAE latent space by feeding it into the encoder. We did the same in the SOM space, this time by matching the elements of the set with the closest SOM chroma class center. We then considered only the 24 centroids of the clouds corresponding to each chord in the 2D space obtained by PCA reduction as in figure 2.

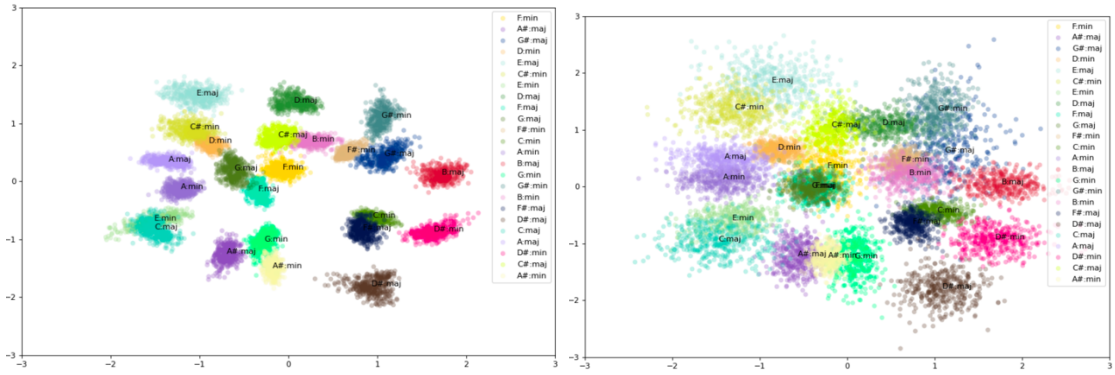

Figure 2. Chords data set projected in the VAE latent space. Variants of the same chord appear as compact clouds. 3dimensional VAE codes reduced to 2D by PCA. 100 learning epochs (left), 299 epochs (right). From (Feldman 2021) report.

We set a measure of compacity *C* to evaluate the latent spaces power of representation, i.e. to what extent close points in the latent spaces do correspond to close elements in the harmonic system. We compute the average euclidean distance

of each point to its K nearest neighbors in the chroma domain (Carsault & al 2018) and average over the whole space to give it a global score *C*. We take as a ground reference the Tonnetz where chords neighbors are related by neo-riemann LPR relations (Cohn 1997) expressing largely accepted common harmonic proximity. Compacity is computed the same way (chroma distance between close neighbors) for the SOM and the Tonnetz. The compacity results are $C = .28$ for the Tonnetz, $C = .42$ for the VAE 3 dimensions latent space (the number of dimensions achieving the best result), and $C = .52$ for the SOM, showing that the VAE is the best learning model so far. The difference with the Tonnetz supposedly optimal result also shows that by training over polyphonic data sets presenting a great variety of pitch combinations, we are able to organize a significant amount of complex textures, but we thereby introduce noise in the latent space, somehow blurring the simple harmonic relations : constructing a latent space that is friendly to arbitrarily complex sonic textures while maintaining optimal coherence with simple harmonic relations is yet an open research issue, so we stick with this compromise for now.

Somax2 In a nutshell

Somax2 agents are loaded with a local stylistic knowledge space (called a Corpus) providing a dynamic musical memory at run time, and weave a "co-improvisation" by navigating their musical memory while being sensitive and reactive to the external context (live musicians or other agents).

Let's say you hear the first notes of the song “Michelle” by the Beatles (or any other melody that you actually know). This automatically activates the memory of that melody, that should then keep singing in your head even in the absence of the external input, lighting up areas of cortical maps and creating brain activation patterns. This simple analogy serves as the basic behavioral model for Somax2. Musical streams unfold over time. Different views can be used to interpret them, for instance as sequences of pitch, mfcc vectors, chromagrams, onsets etc. When an

element or a sub-trajectory is recognized from a stream (e.g. a fragment of a melody, a chord progression, a rhythmic pattern), corresponding areas of the musical memory get activated like hotspots. At all times, the machine tries to 'understand' the current musical situation, which comprises both its own playing and the live environment, under all the different musical views available. Hotspots are considered relevant for the current musical time and map materials that apply as best candidates to be played in order to fit the current musical flow and move it forward.

Navigation strategy

To build a Corpus from musical data in the form of an audio or midi file, the first step is to segment the musical stream into discrete units or «slices» between two event onsets. Each slice is analyzed and classified with regards to a number of musical features, s.a. pitch, texture, timbre, dynamics, rhythm etc., , and these features will serve as the main basis for constructing the navigation model. The textural clustering is done with regard to the latent space chosen, i.e. two slices belong to the same class if their texture (more precisely their chroma content) encodings are close in the latent space.

From the sequence of slices a statistical sequence model is built that expresses the motivic organization (the structure of repetitions and variations). Statistical models used so far are variable markov models (Cheng-I Wang & al 2016) such as Factor Oracles as in Omax (Assayag & Dubnov 2004) or simply collections of n-grams. This is repeated for each feature in the analysis, effectively resulting in a multilayer representation (StreamViews). The statistical model accounts for short term memory and continuity when recombining patterns at generation time. The original musical content along with the metadata generated by the multilayer classification and sequence model are stored as a corpus file. At run time, an agent loads a Corpus file into a dynamic musical memory structure, or navigation model.

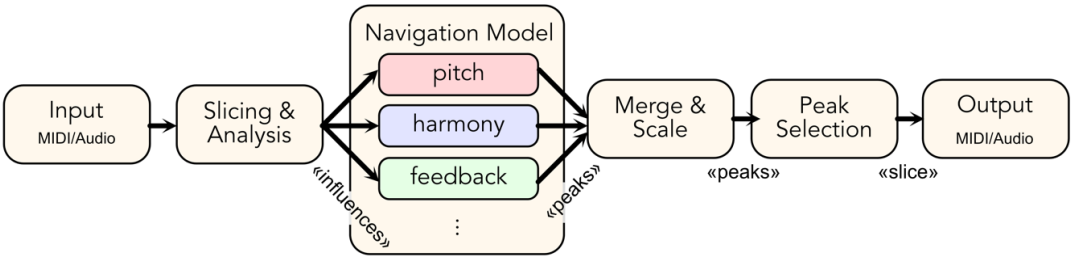

Figure 3. Somax2 basic workflow (in the Midi version), with as example pitch and chromas (harmony or texture) descriptors layers or StreamViews, and the "self" (feedback) layer. Not shown here, the feedback itself subdivides similarly into feature layers.

Influences and activations

When a musician interacts with an agent, their stream goes through the same process of segmentation and multilayer feature analysis and classification, producing a stream of "influences" of different types. Moreover, the stream of input features goes through an echoic memory simulator based on a leaky integrator that simulates the persistence of percepts and "thickens" the textures, and through a sequence model that groups unit "slices" into patterns, so a short term memory of the input stream is always maintained up to a given depth that is a parameter of the system. Influences are matched to the content of the navigation model, generating continuous activations, or «peaks», at certain locations in the musical memory, where the input recent history is similar to memory subsequences ending up at these locations, with regard to features / classes.

Each peak is similar to a narrow gaussian over a location in the memory, so the entire set of peaks can effectively be seen linearly as a one-dimensional continuous influence curve, after peaks from all the feature layers (StreamViews) have been merged by linear combination as in figure 4

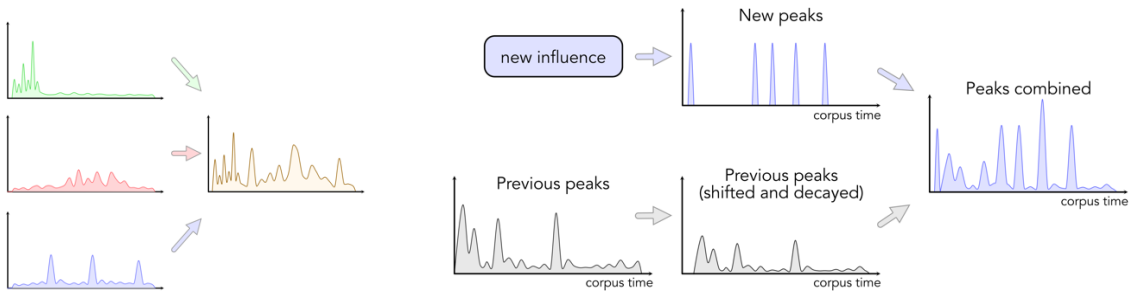

Figure 4. linear combination of influence peak profiles organized by StreamViews (left) and the Shift/Decay/Merge mechanism updating the memory peak profile at each time step (right)

Somax2 updates the memory peak profile at time *t* by merging the current influence curve with the peak profile at time *t-1* shifted by one memory time unit and exponentially decayed. This shift/decay/merge mechanism expresses a form of persistence of the influences, and reinforces the short term sequential model : suppose an influence *A* has raised hotspots at a series of locations $\{A_i\}$ in memory. At next time unit *t+1* an influence *B* arrives and lights up locations $\{B_j\}$. If it happens that there are memory locations where *AB* appear in sequence, the peak in *B* will be enforced by the shifted *A* from time *t*. This of course enforces matches with subsequences matching the input history, moreover this extends the primitive sequential model to a novel fuzzy pattern recognition scheme. In effect, it is easy to see that if ABC is heard at the input, all the ABC-conformant patterns will be reinforced in the memory, but also (and with a lesser gain) more loosely similar ones such as AxC, xBC etc (see figure. 5).

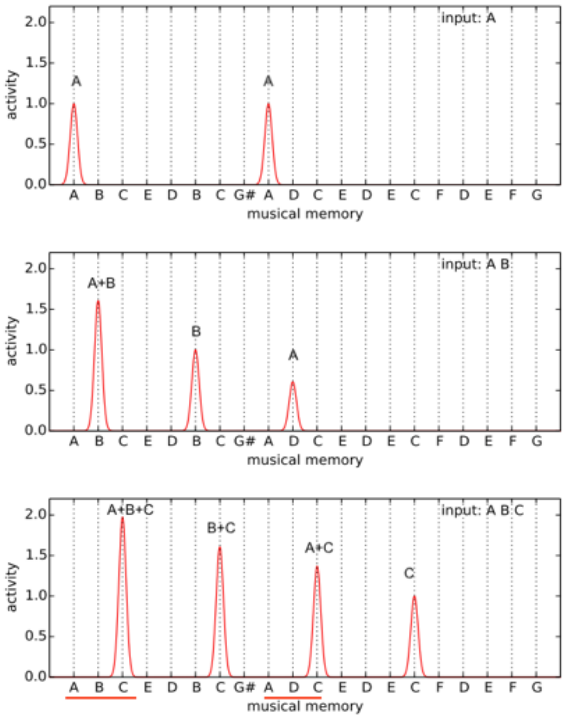

Figure 5. When inputting the sequence ABC, shift/decay/merge mechanism will recognize patterns ABC, but also BC, ADC, C, etc. effectively resulting in a fuzzy pattern recognition.

In addition to "external" streamviews that characterize the external input to a player, there's also a feed-back layer that listens to the generated output of the agent itself. It uses the same influence mechanism and generates peaks at locations "predicted" (by the sequence model) from the recent events played by the agent. So the self-coherence induced by the sequence model, that encourages the player to generate stylistic variations induced by its learned corpus, comes into balance with the sharp reactivity to context, which prompts the player to go into directions that respect and magnify the recent history of external influences. This tends to conform an agent somehow gifted both with a musical ear and a stylistic "culture".

Finally, all the different views are weighted and combined so as to give an activation profile pointing to places in the musical memory that best match the current musical flow. Depending on the relevance of each view (how much they get activated), as well as the weights attributed to each of them, the machine will be showing a more "harmonic" (or melodic, or timbral, or rhythmic etc.) ear. It will

also be more prone to respect its internal logic and follow its own idea, or rather match as closely as possible the current external musical situation and show that it does "listen".

Finally, .. let's Play

Finally the peak distribution in an influence profile, continuously shifting and reshaping with the evolution of the external context and of the navigation/generation process itself is observed as a probability distribution in order to make a decision on the next element to play, by choosing a memory event at a significant peak location, until the request to play.

An important control of interaction dynamics consists in tuning the balance between the different musical dimensions of influences, as well as that between self-influence and sensitivity to external context. The Somax2 GUI gives an extensive set of controls for these and many other parameters (figure 6a).

So the general interaction loop for a player can be summed up as : continuously collect influences coming from all sources (self, input, other players etc.) and update memory peak profile accordingly ; respond to next-event requests by choosing an event in memory and output it. Next-event requests depend on the chosen interaction strategy which is a parameter of the system controllable through the GUI : it may depend on triggers induced by input events recognition based on energy, feature recognition such as pitch or texture estimation, pattern recognition, self auto-play next-event signals, clock signals, etc.

Figure 6. Somax2 usual GUI (right) and a player detailed (left). Inside a player, from top to bottom : source influence routing interface, streamview layers balance for self and external influence, statistical model variable memory length, matches visualisation, output influence interface, rendering interface.

By keeping a parsimonious trace of the different places that were activated and by accumulating evidence for places that are relevant sequentially and in time, the process is able to detect the places in the memory that make more sense in the current musical situation, while taking into account recent past activations, hence proposing a solution to the locality problem (when a system reacts with a limited listening time scope thus missing the long term coherence of the input), and providing a form of anticipation. Typical example of the locality problem is that if you actually play literally the melodic part of a given corpus, the agent should theoretically respond by following the original path and purely replicating the corpus, which does not happen in general.

An important question arising when proposing a generative model is that of evaluation. How do we establish that the Somax2 productions are satisfying ? Of

course, the first intention is to observe human reactions. Are they pleased and stimulated to play with the system, are they ready as professional artists to engage their reputation by performing in major venues with the system, do they consider the musical experience as novel, rewarding, and creative. These questions are handled by the social science research branch of the REACH project using the methods of interviews, collecting artists feed-back, and audience feed-back as well. More deeply, these researches pose the question of social acceptability of new musical forms involving interactive AI on stage in the musical communities concerned. An upcoming publication in the "Cahiers d'ethnomusicologie" by Marc Chemillier and Yuri Prado in the REACH project will address this question in the case of a popular French-Malagasy musician, Charles Kely Zana-Rotsy, who has included AI REACH software in his Malagassy-Funk band line-up for a series of concerts, thus validating the acceptability criteria in this particular situation indeed. On a more computational track, Shomo Dubnov has recently initiated researches in REACH to characterize the info-theoretic transfer of information between a control signal (the human) and a resulting signal (the response of the virtual improviser). Two known measures, the mutual information and the information rate (Dubnov 2021) are combined to propose an estimation of the transfer entropy, whose value could be considered as a marker of the quality of co-articulation between two co-improvisers (Dubnov, Assayag, Gokul 2022). This approach could not only bring progresses in the evaluation question, but also provide a new kind of loss function for deep generative learning of improvised styles.

Somax2 Model

We detail here only the memory activation - selection part of the Somax2 core model.

Memory activation model

The guidance of SoMax is fully conducted according to the *activity* principle. An agent possesses a sequential musical memory constituted by musical content (e.g. an audio buffer, midi data,) and analytical metadata organized in memory views (StreamViews). A memory view is a segmentation of the memory into *m* events $\{event_i\}$ at places $i=1...m$, classified with regard to some general space of representation (called *κ-space*). An example of *κ-space* is the set of pitches, or the textural latent space modeled by the chroma SOM described above. Now let O_t be an observation "heard" either from the external context or from the generation itself. For a given StreamView and associated *κ-space* each place *i* of the memory will be stimulated according to an activity level $\alpha_i \in (0..1)$, which quantifies the similarity between $event_i$ and observation O_t as evaluated through the chosen *κ-space*. This means that a certain class of representation κ_i (e.g. a pitch class, a chroma class, a chroma VAE encoding) from the *κ-space* has been deemed the same for O_t and $event_i$.

For example, with *κ-space* = SOM, αi is related to the Euclidean distance d_i between two chromagrams : the one extracted from O_t and the one at the center of the SOM class that relates O_t and $event_i$: $\alpha_i = exp(-cd_i)$, for some scalar *c*.

By extending this principle to sequences of states, it is possible to obtain an activity depending on the past of the improvisation, by comparing the sequences $O_{t-n}... O_t$ with $\forall i \in)n, m) \kappa_{i-n}...\kappa_i$, i.e. a continuity with a past of n states. In order to facilitate the fast access to these sequences, an efficient statistical sequence model s.a. a Factor Oracle or an n-gram hash-table is built over the memory and associated to the view.

Let be a relative time scale function of time, $\xi \equiv \Psi(t)$ which can be dependent e.g. on a beat/tempo structure (in the case when beat is relevant, the decimal part will represents phase inside the beat).

If a memory place *j* whose corresponding $event_j$ is located at date ξ_j (in memory time) is activated, the activity is modeled as a Gaussian around this date ξ_j. For any other memory date ξ, the contribution of *j* to its own activation level is :

$$\gamma_\sigma(\xi;\, \xi_j, a_j) = a_j \exp\left(-\frac{(\xi - \xi_j)^2}{2\sigma^2}\right)$$

where σ represents an activation time spread, and a_j the activation at j . The total activation of the position ξ in the memory is therefore the sum of these contributions

$$\Gamma(\xi) \equiv \sum_j \gamma_\sigma(\xi;\, \xi_j, a_j)$$

It is important to make the difference here between the two temporal scales that we manipulate : the time of the memory and the time of the performance. Up to now, performance time was “frozen” and we were reasoning on memory time.

Let us stimulate at performance time *t* a place *j* in the memory by an observation *O(t)*. The peak caused by this observation at memory time ξ_j will not just appear and disappear when the external observations context will change. It will rather continuously shift forward in memory time as well as in performance time with an exponential attenuation (as already seen in figure 5). This property can be written:

$$\Gamma_{t+\Delta t}(\xi) = \Gamma_t(\xi - \Delta\xi) \times \exp\left(-\frac{\Delta\xi}{\tau}\right)$$

Where τ is the time decay factor. This corresponds to a principle of perceptive persistence of information similar to echoic memory (Radvansky 2005) : it means that a place in memory, if it does not agree right now with the observation context but experiences a common past with it, is nevertheless perceptually relevant. Its activity level, i.e. its adequacy with the context, is however smaller than a direct match. This approach makes it possible to react to the present of the improvisation, without forgetting the near past, and enriches the strict sequential models with a fuzzy recognition capacity as seen in figure 5. In the end, the improvisation will be guided towards the places of the memory selected through a probability distribution directly derived from the activity profile, sometime choosing

places that are not the best match with the present but that make sense with regards to the motivic memory, thus increasing the creativity of the system.

Memory Selection

Indeed, it is possible to interpret the activity of a place *i* at date ξ as the probability of being selected as the most context-sensitive with respect to current observation of the context ***o***, using the softmax function

$$P(\xi|\mathbf{o}) \equiv \frac{\exp(\beta\,\Gamma(\xi))}{\sum_{\xi'}\exp(\beta\,\Gamma(\xi'))} \qquad (1)$$

where β is a regularization factor. P(ξ|o) can be seen as the probability of choosing ξ as a best match to the current musical flow. Alternatively, this probability can be seen as the probability that everything that has been observed comes from some unknown process that ends at ξ. What follows next in the musical memory is thus a potential candidate for a future match :

$$\hat{\xi} \equiv \underset{\xi}{\operatorname{argmax}}\, P(\xi|\mathbf{o}) = \underset{\xi}{\operatorname{argmax}}\, \Gamma(\xi)$$

Now it is easy to generalize to multiple memory views. Let us consider a case involving two views, corresponding to observations o_1 and o_2 with respective activities Γ_1 and Γ_2 (generalization to more than two views is straightforward). Views o_1 and o_2 might be selected from e.g. external listening / harmonic view, and self-listening, pitch view (the UI lets the user configurate and weight such selections). We want to find the event that best matches the current musical flow, *ie* that maximizes $P(\xi|\, o_1\, o_2)$. Following Bayes rules this can be written as :

$$P(\xi|\mathbf{o}^{(1)}\mathbf{o}^{(2)}) = \frac{P(\mathbf{o}^{(1)}\mathbf{o}^{(2)}|\xi)P(\xi)}{P(\mathbf{o}^{(1)}\mathbf{o}^{(2)})}$$

Assuming independence and conditional independence given ξ of the observations and using Bayes' rule one more time yields:

$$P(\xi|\mathbf{o}^{(1)}\mathbf{o}^{(2)}) = \frac{P(\mathbf{o}^{(1)}|\xi)P(\mathbf{o}^{(2)}|\xi)P(\xi)}{P(\mathbf{o}^{(1)})P(\mathbf{o}^{(2)})}$$

$$P(\xi|\mathbf{o}^{(1)}\mathbf{o}^{(2)}) = P(\xi)^{-1}P(\xi|\mathbf{o}^{(1)})P(\xi|\mathbf{o}^{(2)})$$

Assuming a uniform prior, we can write $P(\xi)^{-1}$ as a constant K. Taking the log and using (1) thus gives:

$$\begin{aligned}\log P(\xi|\mathbf{o}^{(1)}\mathbf{o}^{(2)}) &= -\log K + \beta_1\Gamma_1(\xi) - \log\left(\sum_{\xi'}\exp\left(\beta_1\,\Gamma_1(\xi')\right)\right) \\ &\quad +\beta_2\Gamma_2(\xi) - \log\left(\sum_{\xi'}\exp(\beta_2\,\Gamma_2(\xi'))\right)\end{aligned}$$

Given that we are only interested in the argmax, we finally have:

$$\log P(\xi|\mathbf{o}^{(1)}\mathbf{o}^{(2)}) \propto \beta_1\Gamma_1(\xi) + \beta_2\Gamma_2(\xi)$$

$$\underset{\xi}{\operatorname{argmax}}\, P(\xi|\mathbf{o}^{(1)}\mathbf{o}^{(2)}) = \underset{\xi}{\operatorname{argmax}}\,(\beta_1\Gamma_1(\xi) + \beta_2\Gamma_2(\xi))$$

In other words, considering total activity Γ as the weighted sum of the two activities Γ_1 and Γ_2 is actually motivated by Bayesian considerations. This is reminiscent of (and inspired by) multimodal sensory cue integration as in (Landy et al., 2011; Fetsch et al., 2012). The weight given to each of the views can be seen as the confidence in the relevance of the corresponding view. This concept of multiple activity is powerful because it allows us to extend the dimensionality of our reactive listening and to weigh the contribution of each viewpoint specific to different influence sources (external, self, players) and to various musical features (harmony, texture, melody, rhythm, ..). In addition, activity time is modeled continuously even though the memory's event structure is discrete and sequential, so our different viewpoints do not need to be fragmented at the same granularity (ie event segmentation of the same musical content can be different from a viewpoint to another), which fits well musical reality where different dimensions progress at different paces (e.g. harmonic rhythm is slower than melodic rhythm).

Heuristic guidance.

It is also possible to guide Somax2 through a "heuristic lead" by artificially generating activity patterns that will constrain the navigation to serve a given musical purpose. For example predefined scenarios could be approximated by activating locations that match the scenario's elements in an appropriate κ-space (e.g. soloing over a given harmony could be achieved by enforcing activity in the locations that match the current chord). We use in particular heuristic guidance to adapt dynamically Somax2 to pulse based context. Providing the corpus includes beat markings and the external signal has a pulsative nature, a memory view can adapt its clock to the external observations stream and try to continuously align to the external beat (using beat-tracking, a tap tempo or otherwise). Before selecting a location to jump to, the view generates a heuristic activity profile in the shape of a cosine with a period equal to the current beat duration and with a phase equal to the current position within the beat (shared between memory time and performance time) .

$$\Gamma_{phase}(\xi) = \Gamma(\xi) \times \exp\left(\eta[\cos\left(2 * \pi(\xi - \xi_{target})\right)] - 1\right]$$

This will create strong activity peaks at positions $\{\xi_{beat\text{-}phase\text{-}i}\}$ in the memory that fall at the correct beat/phase place, thus making sure that the next jump will not violate the beat alignment. In the case $\xi_{beat\text{-}phase\text{-}i}$ does not correspond to an event boundary, some heuristics has to be employed in order to keep the coherence of the generation stream. As possible solutions, a filter can be applied to select only dates ξ_j that are close enough to an event boundary, ξ_e then account for the offset $\delta = \xi_j - \xi_e$ then try to resorb this difference, either by slightly acting on the scheduler, or by reabsorbing it in the next jumps.

Somax2 Architecture

We give some details here on the concrete implementation. Somax2 is designed in a modular way as a framework composed of two main components: (a) a server, written in Python, handling the offline construction of the corpus as well as the real-time mapping and scheduling occurring in the navigation model described above,

and (b) clients (possibly running on different machines), written in MaxMSP (but other clients may be developed in other environments), handling the real-time I/O, the GUI, and the audio and midi rendering (figure 7). Each of these clients has a highly modular design, allowing the user to freely recombine the modules and easily extend them to implement new behavior. Main objects in the Max front-end have their own GUI and they communicate with a python back-end model through an OSC communication channel.

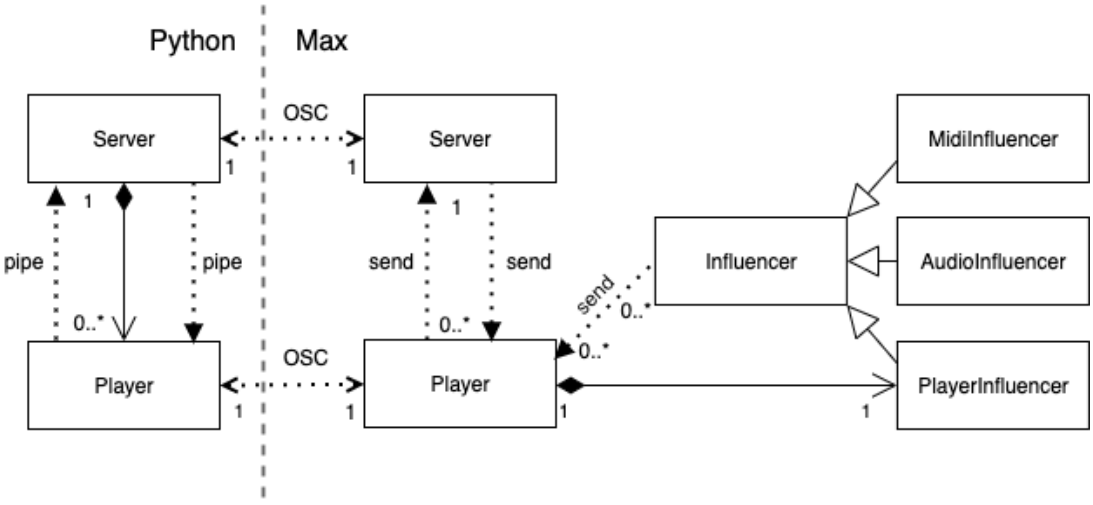

Figure 7a.The client / server architecture of Somax2

Max Front End architecture

The Max architecture currently consists of four main objects: AudioInfluencer, MidiInfluencer, Player and Server. The AudioInfluencer and MidiInfluencer read from external inputs a continuous stream of audio or MIDI data respectively and perform the slicing and feature analysis steps described above. In the front end, the data resulting from this process is called an influence, which has to be routed to a player (and its Python back-end model) that will compute the following steps in the influencing process. The Player object is essentially a client for the corresponding Player class in the Python back-end, which handles all of the agent runtime architecture. Finally, the Server object handles communication with the Server class

in the Python back-end, which is the root of the entire system, handling all players models as well as the transport (i.e. the master clock of the event scheduling). A simplified diagram of the entire system can be seen in figure 7a, and a display of the same workflow through within the Max API in figure 7b.

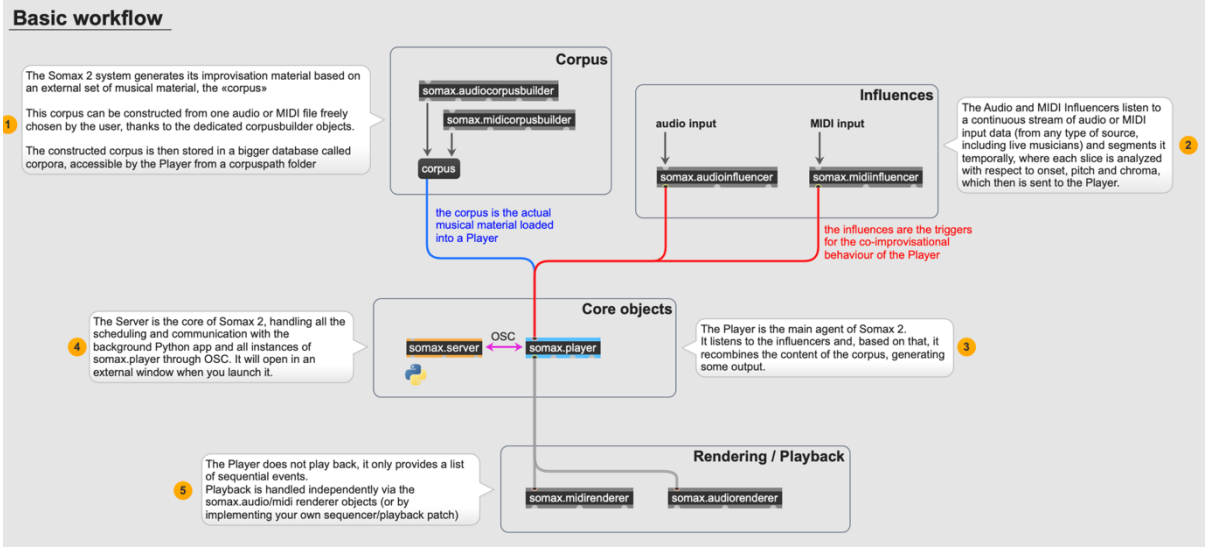

Figure 7b. Somax2 basic workflow. Somax2 can also be programmed directly as a Max library to build one's own application and GUI (tutorial by Marco Fiorini)

The Player object in Max in addition to its main role as a reactive generator, has been given two additional roles: routing and re-influencing. The routing module of the player (whose user interface is depicted in figure 8) lists all available influencers and players and receives influences based on what the user selects. The player also contains an influencer module (PlayerInfluencer) so it can itself act as a source of influence to other players. This way, arbitrarily complex interconnected networks of players / input modules can be setup and reshaped dynamically, also allowing circular relationship (Player A listens to Player B who listens to Player A, creating a form of "homeostatic" equilibrium).

Figure 8. The routing module of each player. Here, all influencers and other players are listed, and the user can select whether to listen to pitch influences (P), onsets (O) and/or chroma influences (C) from that particular source. As chroma influences are continuous, it's also possible to give weight to chromas from different sources.

A Model-View-Controller design pattern has been applied to ensure that no inconsistencies exist between the Python back-end (model) and the Max user interface (view and controller). Each object has been modeled so that multiple views/controllers may exist for the same data, so the GUI complexity can be tailored to everyone's need.

Python Back End Architecture

The Python back-end is implemented as a parallel multicore architecture. This solution was chosen since an influence-generate cycle for a single player can in extreme cases take 30 milliseconds or more. This is still manageable providing the latency stays fairly constant. However this latency would cumulate for multiple concurrent players on a single core leading to unacceptable temporal unevenness.

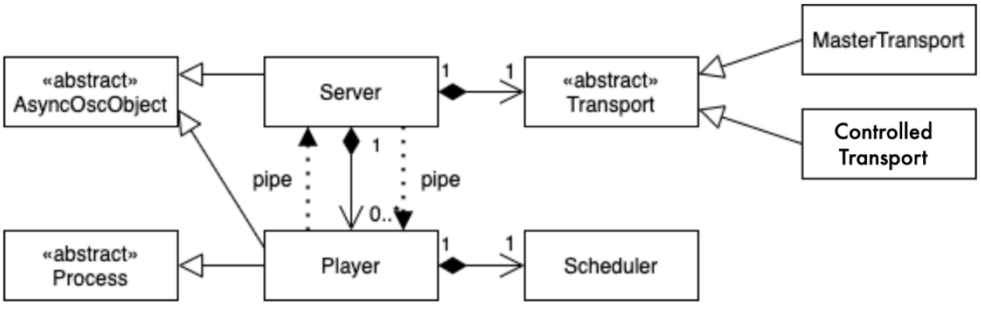

Figure 9. Simplified class diagram over the Python component architecture

Each Player runs in its own Process, thereby not impacting each other in terms of

performance, as long as there are fewer players running than free cores on the system. The Server is running two coroutines using the python asyncio module, where one coroutine continuously receives messages from the (client) Max Server object over OSC and the other coroutine continuously updates the time of the scheduler Transport class and forwards the time to each Player. The controlled transport is intended for the case when an external time source is to take control (e.g. Ableton Live, a Max patch etc.)

The Player class is also running two coroutines; one coroutine continuously receiving messages from the Max Player object (parameter updates, influences and onsets triggers) and one coroutine continuously receiving messages from the pipeline connected to the Server. These messages are queued through each Player's scheduler (figure 9).

Runtime architecture

The runtime architecture (figure 10) handles all the influencing and output generation, it's basically the core of the system. It can be seen in figure 9. At the root of the system is the Player class, through which all interaction with the system occurs. At the opposite end is the Atom, where each Atom corresponds to one of the $r = 1, \ldots, R$ layers. The Atom contains one Classifier instance, corresponding to a classifier $\Theta(r)$, one MemorySpace instance, corresponding to a model $M(r)$ and one ActivityPattern instance, which handles storing, shifting, decaying and concatenation of peaks $P(r)$. Inbetween the Player and the Atoms is the StreamView class. Each Player contains any number of StreamViews, which in turn is a recursive structure containing any number of StreamViews and any number of Atoms, effectively forming a tree structure where the Player correspond to the root of the tree, each StreamView correspond to a branch and each Atom correspond to a leaf of the tree. Each Atom and StreamView is assigned a weight $\alpha(r)$ (controlled by the user in the GUI) and at each branch in the tree, merging and user-controlled

filtering are performed by the MergeAction

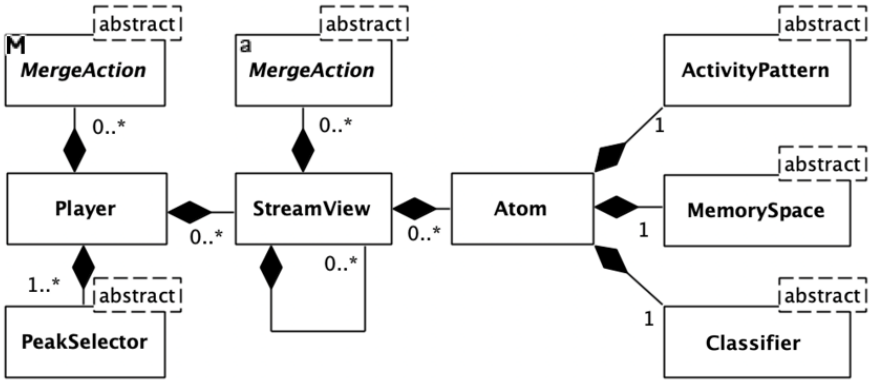

Figure 10. Interaction architecture. Dotted arrow lines denote "wireless" communication between objects while filled arrow lines denote their traditional UML relations (composition, generalization) and corresponding cardinality.

Conclusion

The Somax2 co-improvisation environment has been developed as the last descendant in date of the successful seminal Omax system based on real time style modeling and interaction. It brings radically new features such as strong reactivity based on machine learning and a cognitive memory dynamic interaction model. It takes place in the framework of a large European Project, REACH, that addresses co-creativity between human and machines. Somax2 has shown during experimentations with world-class musicians that it is ready from the outset to interact with them and provide them, without any preparation on their side, with renewed playing interest. Somax2 is already scheduled to perform on stage with world-class musicians in mainstream concert venues and festivals.

Aknowledgements and legacy

Somax2 has been created by Gerard Assayag and Laurent Bonnasse-Gahot, as one of the descendant of the seminal Omax system. Omax was created by the "Omax Brothers" : Gérard Assayag, Georges Bloch, Marc Chemillier and Shlomo

Dubnov, with latest version developed by Benjamin Lévy. It followed seminal researches on style modeling by Assayag, Dubnov and Chemillier at Ircam.

Somax2 current development is by Joakim Borg, following earlier implementations by Laurent Bonnasse-Gahot and Axel Chemla-Romeu-Santos. Mikhail Malt and Marco Fiorini are also strong scientific and artistic contributors.

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (Grant agreement #883313) and from the Agence Nationale de la Recherche (MERCI ANR-19-CE33-0010).

References

Gérard Assayag, S. Dubnov « Using Factor Oracles for machine Improvisation », Soft Computing, vol. 8, n° 9, Septembre, 2004

Gérard Assayag, G. Bloch, A. Cont, and S. Dubnov. Omax brothers : a dynamic topology of agents for improvisation learning. ACM Multimedia, 2006.

Gérard Assayag, Improvising in Creative Symbolic Interaction, in J. B. L. Smith; E. Chew; G. Assayag (Eds) Mathematical Conversations : Mathematics and Computation in Music Performance and Composition, World Scientific; Imp. College Press, 61 - 74, 2016

Gérard Assayag, Co-créativité humains-machines. Une réflexion sur les indisciplines symboliques. Revue Francophone d'Informatique et Musique, AFIM, 2020.

Gérard Assayag, Human-Machine Co-Creativity, in B. Lubat, G. Assayag, M. Chemillier (Eds). Artisticiel / Cyber-Improvisations. Phonofaune, Dialogiques d'Uzeste 2021.

Bednar J.A., Wilson SP. Cortical Maps. The Neuroscientist. 2016, 22(6):604-617.

Blackwell, T., Bown, O., & Young, M. (2012). Live algorithms: Towards autonomous computer improvisers. In J.McCormack, & M. d'Inverno (Eds.), Computers and creativity (pp. 147–174). Berlin: Springer.

Georges Bloch, Jérôme Nika. Edith Piaf, Billie Holiday, and Elisabeth Schwarzkopf Making Music Together. Kapoula Z., Volle E., Renoult J., Andreatta M. Exploring Transdisciplinarity in Art and Sciences, Springer, pp.275-299, 2018

Jean-Pierre Briot, Gaetan Hadjeres, and Francois Pachet. 2019. Deep Learning Techniques for Music Generation. Springer, series Computational Synthesis and Creative Systems.

Tristan Carsault, J. Nika, P. Esling, G. Assayag. Combining Real-Time Extraction and Prediction of Musical Chord Progressions for Creative Applications. Electronics, MDPI, 2021, 10 (21), pp.2634.

Chadebec, C. and Allassonnière, S., A Geometric Perspective on Variational Autoencoders, Neural Information Processing Systems (NeurIPS 2022).

Marc Chemillier Jazz and Artificial Intelligence : From Presence to Traces. In B. Lubat, G. Assayag, M. Chemillier (Eds). Artisticiel / Cyber-Improvisations. Phonofaune, 2021,

Conklin, Darrell. (2003). Music Generation from Statistical Models. Journal of New Music Research 45:2.

R. Cohn. Neo-riemannian operations,parsimonious trichords, and their" tonnetz" representations. Journal of Music Theory, 41(1):1–66, 1997.

Ken Déguernel, Emmanuel Vincent, Gérard Assayag. Probabilistic Factor Oracles for Multidimensional Machine Improvisation. Comp. Music Journal, 2018, 42 (2), pp.52-66.

Ken Déguernel, E. Vincent, J. Nika, G. Assayag, K. Smaïli. Learning of Hierarchical Temporal Structures for Guided Improvisation. Computer Music Journal, 2019, 43 (2)

Diederik P. Kingma and Max Welling (2019), "An Introduction to Variational Autoencoders", Foundations and Trends® in Machine Learning 12: 4, pp 307-392.

Hao-Wen Dong, K. Chen, S. Dubnov, J. McAuley and T. Berg-Kirkpatrick, Multitrack Music Transformer, ICASSP 2023 - 2023 IEEE, Rhodes Island, Greece, 2023, pp. 1-5

Shlomo Dubnov. Machine Improvisation in Music: Information-Theoretical Approach. In: Miranda, E.R. (eds) Handbook of Artificial Intelligence for Music. Springer 2021

Shlomo Dubnov, G. Assayag, V. Gokul Creative Improvised Interaction with Generative Musical Systems IEEE Intl. Conf. on Mult. Inf. Proc. and Ret. (IEEE-MIPR) 2022 121-126

Shlomo Dubnov, V. Gokul, G. Assayag. Switching Machine Improvisation Models by Latent Transfer Entropy Criteria. Physical Sciences Forum, 2023, 5 (1), pp.49.

Shlomo Dubnov, K. Chen, and K. Huang. Deep music information dynamics: Novel framework for reduced neural-network music representation with applications to midi and audio analysis and improvisation. Journal of Creative Musical Systems, 1(1), 2022.

Fetsch, C., Pouget, A., DeAngelis, G. et al. Neural correlates of reliability-based cue weighting during multisensory integration. Nat Neurosci 15, 146–154 (2012).

Benjamin Feldman, Improving the latent harmonic space of SOMax with Variational Autoencoders, Master's report, Ircam, 2021; G. Assayag and T. Carsault supervisers.

Théo Golvet, L. Goupil, P. Saint-Germier, B. Matuszewski, G. Assayag, et al. With, against, or without? Familiarity and copresence increase interactional dissensus and relational plasticity in freely improvising duos. Psychology of Aesthetics, Creativity, and the Arts, American Psychological Association, 2021

Teuvo Kohonen, "The self-organizing map," in Proceedings of the IEEE, vol. 78, no. 9, pp.

1464-1480, Sept. 1990, doi: 10.1109/5.58325.

Teuvo Kohonen, Essentials of the self-organizing map, Neural Networks, Volume 37, 2013, pp 52-65, doi.org/10.1016/j.neunet.2012.09.018.

Landy, M. S., Banks, M. S., and Knill, D. C. (2011). Ideal-observer models of cue integration. Sensory cue integration, pages 5–29. Nature Neuroscience, 15:146–154.

Georges Lewis, Too many notes: Computers, complexity and culture in voyager, Leonardo Music Journal 10, 33-39, 2000

George Lewis, Co-Creation: Early Steps and Future Prospects., In B. Lubat, G. Assayag, M. Chemillier (Eds). Artisticiel / Cyber-Improvisations. Phonofaune, 2021.

Maxwell, James, A. Eigenfeldt, P. Pasquier & N. Thomas (2012). Musicog: A cognitive architecture for music learning and generation. Proc. of the 9th Sound and Music Computing Conference, SMC 2012.

Jérôme Nika, Marc Chemillier, Gérard Assayag. ImproteK: introducing scenarios into human-computer music improvisation. ACM Computers in Entertainment, 2017

Jérôme Nika, K. Déguernel, A. Chemla-Romeu-Santos, E. Vincent, and G. Assayag. Dyci2 agents: merging the "free", "reactive", and "scenario-based" music generation paradigms. International Computer Music Conference, Shangai, China, Oct 2017.

Radvansky, Gabriel (2005). Human Memory. Boston: Allyn and Bacon. pp. 65–75.

Tatar, Kıvanç & Pasquier, Philippe. (2017). 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).

Tatar, Kıvanç & Pasquier, Philippe. (2018). Musical agents: A typology and state of the art towards Musical Metacreation. Journal of New Music Research. 47. 1-50.

Toiviainen, Petri. (1997). Optimizing self-organizing timbre maps: Two approaches. In: Leman, M. (ed) Music, Gestalt, and Computing, LNCS vol 1317 Springer, Berlin

Toiviainen, Petri. (2005). Visualization of tonal content with self-organizing maps and self-similarity matrices. Computers in Entertainment. 3. 1-10

Cheng-I Wang, J. Hsu, S. Dubnov, Machine Improvisation with Variable Markov Oracle: Toward Guided and Structured Improvisation. Comput. Entertain. 14(3): 4:1-4:18 (2016)

Norbert wiener, Cybernetics or control and communication in the animal and the machine, 1948. New York: John Wiley & Sons, Inc. Paris: Hermann et cie.

Geraint A. Wiggins, Searching for computational creativity. New Generation Computing, 24(3):209–222, 2006.