Journal of New Music Research

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/nnmr20

Human–machine agencies in live coding for music performance

Anna Xambó & Gerard Roma

To cite this article: Anna Xambó & Gerard Roma (31 Dec 2024): Human–machine agencies in live coding for music performance, Journal of New Music Research, DOI: 10.1080/09298215.2024.2442355

To link to this article: https://doi.org/10.1080/09298215.2024.2442355

Published online: 31 Dec 2024.

JOURNAL OF NEW MUSIC RESEARCH
https://doi.org/10.1080/09298215.2024.2442355

Routledge
Taylor & Francis Group

FORTHCOMING IN THE SPECIAL ISSUE: TECHNOSCIENTIFIC PRACTICES OF MUSIC; CRITICAL IMPLICATIONS OF NEW TECHNOLOGIES

Human–machine agencies in live coding for music performance

Anna Xambó[a] and Gerard Roma[b]

[a]Centre for Digital Music, Queen Mary University of London, London, UK; [b]School of Computing and Engineering, University of West London, London, UK

ABSTRACT

In this article, we explore the notion of human–machine agencies in live coding applied to music performance. We first survey the literature on interactive music systems, new interfaces for musical expression, live coding and human–data interaction to identify common characteristics related to agency, which can be applied to the analysis of live coding practices. Then, we propose a theoretical framework for supporting the design, evaluation, and analysis of data-driven, AI-enhanced live coding systems. The framework, inspired by actor-network theory and human–data interaction, is composed of four dimensions: *legibility*, *modifiability*, *predictability* and *cardinality*. We reflect on two live coding systems built by the authors, utilising the four dimensions. We hope that the framework will help understand live coding in the datafication era, in which the data becomes a participating agent. Beyond live coding, our analysis can inform other artistic practices that use AI in interactive or real-time settings while keeping the human in the loop.

ARTICLE HISTORY
Received 3 July 2024
Accepted 10 December 2024

KEYWORDS
Actor-network theory; AI; human–data interaction; human–machine agency; interactive music systems; live coding; machine learning; NIMEs

1. Introduction

The recent popularisation of artificial intelligence (AI) technologies has brought an urge to understand and explain the relationship between human and machine agencies in artistic disciplines such as music. In this article, we propose that the practice of live coding in computer music performance offers a fertile ground for understanding and experimenting with human–machine agencies. The focus on the agencies emerging from the human–machine relationship can inform other artistic practices that use AI. Drawing from actor-network theory (ANT) (Latour, 2005), this article assumes that agents, both humans and non-humans, are equally relevant in the interplay of action.

The notion of human–machine agency can be seen as the centre of human–computer interaction (HCI), particularly when AI tools are involved. Suchman (1987) introduced *situated action* as a term describing how users act in a particular context (in Suchman's case when using a photocopier), where shared meanings are constructed according to the situation, depending on the people involved and the technology used. The learning of how the photocopier works happens in a situated action: action and knowledge are interrelated, and collaborative learning happens in the course of action. Suchman made the point that research on machine intelligence should account for the situatedness of human social behaviour for developing a successful human–machine communication. In contrast with planned actions, the concept of situated action emphasises the improvisational nature of everyday interactions with technology. This applies particularly to music and other creative activities where technology is used.

Chadabe (1984) used the term *interactive composing* in the 1980s to refer to the use of real-time computer music systems when composing and performing music. This process was described as 'simultaneously composing and performing by interacting with that system as it functions' (Chadabe, 1984, p. 23). The instrument is perceived as intelligent because it responds to a performer in 'a complex, not entirely predictable way, adding information to what a performer specifies and providing cues to the performer for further actions' (Chadabe, 1984, p. 23). This seminal vision of a 'mutually influential, interactive relationship' (Chadabe, 1984, p. 23), and placing 'a performer in an unusually challenging performing environment' (Chadabe, 1984, p. 23), characterises the entanglement between human and machine agencies in computer-mediated music composition and performance. In this article, we use the term *human–machine agencies* to reflect on the multiplicity of actors involved in this process. We apply this concept to the analysis of music live coding.

CONTACT Anna Xambó a.xambosedo@qmul.ac.uk Centre for Digital Music, School of Electronic Engineering and Computer Science, Queen Mary University of London, Mile End Road, London E1 4NS, UK

Music live coding is a branch of algorithmic music suitable for live performance and music improvisation, in which a musician writes code that produces music. As a tradition with very established practices and principles (Blackwell et al., 2022), music live coding can be easily analysed in the spirit of ANT. ANT can be described as a methodology for studying social phenomena that puts the emphasis on the associations between different actors, where actors may be humans, objects, ideas or processes. ANT is thus well suited for understanding the multiplicity of agencies within techno-social situations. From this point of view, a live coding performance emerges from a network of agents, human and non-human, and the relationships between them. These agents include, among others, the audience and venue, one or more performers, projector and screen, laptops and other music hardware, programming languages, coding environments, and a number of software music processes, with different degrees of autonomy, spawned by the performer(s) coding activities. Note that given the frequent use of the word 'agent' in technical literature and the focus on agency in this article, we will interchangeably use the words 'agent' and 'actor' through the rest of the text.

The concept of *agency networks* was proposed by Brown (2016) as a promising tool to understand musical creativity in musical practices. In this article, we do not attempt to define or explain music creativity applied to human–machine agency. We also refrain from assigning different 'amounts of agency' to different actors. Instead, we propose a set of qualitative dimensions that help explain at least some of the action that takes place in a live coding performance and the role that data can play as part of this complex interaction network. In turn, we hope that these dimensions provide an insight into human–machine agencies that can be used in the design of novel data-driven, AI-powered music live coding tools and environments while keeping the human in the loop.

2. Human–machine agency in music performance

In this section, we chronologically survey the literature on human–machine agency in the areas of interactive music systems, new interfaces for musical expression (NIMEs), live coding, and human-data interaction. A commonality is the search for a range of aspects that describe how human and non-human agents interact together in a shared situation, in this case, the creative act of music-making in music performance.

2.1. Interactive music systems

The origins of musical machine agency can be traced back to the first mechanical devices for playing music, such as automatic carrillons (Roads, 1996). Automatic sequencers and early drum machines were thus naturally implemented in analog keyboards and synthesisers. From these early implementations, the idea of *automatic accompaniment* emerged as a paradigm of musical human–machine agency that can still be observed in much of today's computer-assisted popular music. Automatic accompaniment may make use of different techniques for tracking the action of a human performer, such as pitch or beat tracking (Rowe, 2004).

Machine listening has more widely played a role in interactive music systems. Rowe (2004) explored the idea of improvising and performing with computational agents that could be programmed to respond to MIDI or audio data from the human performer's actions. Another example of early interactive music was GenJam (Biles, 1994), which developed the use of genetic algorithms for live improvisation. The Continuator (Pachet, 2003) proposed interactive music performance as a dialogue with an agent that learns to imitate the human's style.

From these early examples, it can be seen that the paradigm of automatic accompaniment was extended to consider autonomous computational agents playing along with human performers. Interactive music research has seen significant development into the different roles and capabilities of these agents, notably in computational creativity research.

The application of machine learning (ML) to music improvisation has been a theme of interest for several scholars. Smailis et al. (2021) analysed the OMax, ImproteK, and Djazz systems, developed during the last two decades. These systems allow human performers to explore real-time co-creative improvisation with computational agents. Déguernel et al. (2019) demonstrated examples in jazz music improvisation. A related tradition is based on the Variable Markov Oracle (Wang & Dubnov, 2015), which supports co-creative improvisation using audio databases. A subsequent system proposed by Arias et al. (2016) focused on constructing improvisation scenarios from an audio recording, combining style learning and interactive scenario manipulation.

Some attempts at ranking or classifying systems have been made. For example, Agres et al. (2016) proposed assessing systems according to different *levels of creativity* from *merely generative* to *fully reflective*. Tatar and Pasquier (2019) developed a typology of musical agents based on different criteria, such as *architecture*, *musical task*, *number of roles*, or *human interaction modality*. According to the later criterion (particularly relevant to human–machine agency), agents were classified into *learning from humans*, *controlled by humans* and *playing with humans*.

While a good deal of research on interactive music has focused on understanding the machine side of human–machine agencies, research on the human side is less common. Musicians devise their setups, often in a composition-performance continuum, by either choosing and configuring tools made by others, or by creating their own. The role of the human musician is typically defined reflectively during this process. At the same time, the possibilities for human agency are influenced by the capabilities offered by the tools. Dahlstedt (2021) proposed using several *spectra* (spectrum of *tool complexity*, spectrum of *agency*, spectrum of *generativity*) to help understand human–machine agencies in AI-based musicking with an emphasis on the human side.

2.2. NIMEs

In recent years, we have observed an increased use of ML and deep learning (DL) algorithms in the design and evaluation of NIMEs (Jourdan & Caramiaux, 2023). For example, researchers have incorporated trained ML models in embedded devices, either by augmenting acoustic instruments (DeSmith et al., 2020; Macionis & Kapur, 2018) or by creating new digital musical instruments (DMIs) (Fiebrink & Sonami, 2020; Næss & Martin, 2019). There is also a range of ML toolkits for music, which allow practitioners to create their DMIs based on training their ML models from parameter mappings, such as notably Wekinator (Fiebrink et al., 2009), ml.lib (Bullock & Momeni, 2015), FluCoMa (Tremblay et al., 2021), and Learner.js (McCallum & Grierson, 2020). ML toolkits for music have been instrumental in bringing ML concepts to the NIME community.

Fiebrink and Caramiaux (2018) proposed a human-centred approach to using ML algorithms as a creative musical tool by adapting the HCI concept of interactive machine learning (IML) (Fails & Olsen Jr, 2003). IML incorporates a human-driven interaction between humans and machines that allows users to quickly generate ML models in an interactive setting using small datasets. Fiebrink and Sonami (2020) report the result of several years of co-designing a tool from an IML perspective between a computer scientist/musician and an instrument builder/performer. IML has also been used to produce multi-user models in mobile music by Roma et al. (2018).

Jourdan and Caramiaux (2023) provide a survey of the use of ML in the design of NIMEs from 2012 until 2022. An analysis of 69 papers is done by looking at three aspects: (1) the techniques used; (2) the possibility of interaction between the author/user and the ML system; and (3) the goals of the authors in using ML techniques. The second aspect concerns the interaction between the human and the system, which is categorised into (a) *modifiable parameters* (to what extent the user can change the parameters); (b) *trainable by user* (the extent to which the user can train the model with their own data); (c) *user intervention* (to what extent the user intervened in the process of designing and conceiving the ML system at the beginning, middle or end of the design process); and (d) *evaluation* (how the system is evaluated using quantitative and/or qualitative metrics).

In contrast with traditional ML workflows, Jourdan and Caramiaux stressed the interest of the NIME community in increasing the potential for human intervention, leading to more inclusive systems. Examples of human-centric ML in the design of NIMEs include keeping the human in the loop (Amershi et al., 2014; Dudley & Kristensson, 2018), working with small datasets for the user to maintain control over the system (Fiebrink & Caramiaux, 2018), and incorporating design principles of explainable AI (XAI) (Bryan-Kinns, 2024).

Morrison and McPherson (2024) discuss the concept of *entanglement HCI*, 'the interconnectedness of humans, things, social and ecological systems' (Morrison & McPherson, 2024, p. 3) and its relevance for NIME design. The limits of creative agency are discussed as part of the design constraints of the musical instrument. A mutual negotiation is established between the instrument and the performer, leading to questions such as who controls who, with respect to the human and the technology. The authors propose that in digital instrument design, the post-phenomenological notion of *alterity relation*, in which the instrument embeds generative musical actions and adopts a human-like collaborative role, is of particular interest to NIME.

Several NIME examples that use AI demonstrate the importance of negotiation when collaborating with intelligent music systems. For instance, RAW (Erdem & Jensenius, 2020) is described as a collaborative body-machine instrument that sonifies electromyography (EMG) signals using two Myo armbands and a supervised Multi-Layer Perceptron architecture. The authors describe interaction dynamics by reporting that 'any action can be altered by the subsequent energy influxes from other agencies, be that of a performer, the audience, or a machine' (Erdem & Jensenius, 2020, p. 478) and discuss the exploration of the boundary between control and uncontrol in the search for unpredictability as a creative strategy. Both the interaction dynamics and the exploration of control/uncontrol boundaries concern the extent to which the performer has influence or can be influenced.

Murray-Browne and Tigas (2021) distinguish between traditional ML and DL, advocating for the latter as a suitable tool for creators who want to freely explore

potential mappings between inputs and outputs by navigating latent spaces and latent mappings. Al-terity is a tangible musical instrument with an audio synthesis module that uses a GANSynth DL model for generating audio samples (Tahiroğlu et al., 2020). Tahiroğlu et al. discuss the interaction with Al-terity as a *digital idiomaticity*, with the 'possibility to engage in an unusual state of playing in which the performer remains in a state of a continuous uncertainty' (Tahiroğlu et al., 2020, p. 341).

Many NIMEs include the development of an object in which the physical affordances are clear but the use of AI introduces some degree of uncertainty (Erdem & Jensenius, 2020; Tahiroğlu et al., 2020). In live coding, the tangibility of the code is less obvious, but still several relationships between the performer and the machine are established.

2.3. Live coding

The property of liveness is discussed by Tanimoto (2013), contextualised in programming environments as the ability to modify a running program. This theoretical framework looks into the relationship between the programmer's actions and the computer's responses. Tanimoto outlines six levels (L1–L6) in the process of programming, namely: *informative* (L1); *informative and significant* (L2); *informative, significant and responsive* (L3); *informative, significant, responsive and live* (L4), *tactically predictive* (L5) and *strategically predictive* (L6).

The last two levels of Tanimoto's liveness introduce human–machine agency from the perspective of relationships between programmer actions and computer responses. In L5, the system is slightly ahead of the programmer by predicting the programmer's next action utilising ML techniques, whereas L6 is foreseen as strategically predictive, which indicates more intelligent predictions about the programmer's intentions. Intelligent predictions are linked to agency and liveness, where not only the code but also the tool entails liveness:

> The incorporation of the intelligence required to make such predictions into the system is an incorporation of one kind of agency – the ability to act autonomously. Agency is commonly associated with life and liveness. (One might argue that here, liveness has spread from the coding process to the tool itself.) (Tanimoto, 2013, p. 34)

Xambó (2022) presents a review of past, present and future directions of virtual agents (VAs) in live coding. Two dimensions are introduced to analyse several examples of VAs in live coding: *social interactivity* (does the VA cooperate with other agents, either human or virtual? Is the collaboration product of individual contributions with highly interdependent, mutual interaction or the sum of individual contributions with independent processes?) and *learnability* (does the AI learn, either online during the performance time or offline during pre-performance time?) These categories help understanding how different live coding systems incorporate AI.

Paz and Knotts (2022) assume that live coders who integrate live coding and ML establish a conversational interaction with algorithmic processes, and focus on the 'moments of intervention' in the ML model and workflow. Accordingly, a framework for analysis is proposed based on the axes of *system autonomy* and *performative intervention* in ML processes: the more autonomous the system is, the less performative intervention will be required from the performer. As described later, the modifiability of agent behaviour can still allow for human performative intervention on autonomous agents. Paz and Knotts (2022) tackle the human–machine agency in performance from the perspective of the human-algorithm relationship framed by the ML workflow. Here, we are interested in the property of liveness of the human–machine interaction, i.e. the conversational interaction of the human live coder with the algorithms, and how this shapes the performance.

Rutz (2016) explores the concept of algorithmic agency, or *algorithmicity*, and the relevance of crafting or tuning the models or algorithms assuming that coding practices have a dynamic and experimental nature. Algorithmic agency is the outcome of the mutual human–machine incursions expressed in the writing process. Writing code can be conceived as an entanglement between human and machine, form and medium. Rutz poses questions such as what are the boundaries between human and machine, what are the consequences for authorship and intention, what is the structure of decision-making processes, and what are the mechanisms of control.

Blackwell et al. (2022) pointed out that human–machine relations are an interesting research space for live coding. If we zoom out to the broader picture of the social and political dimensions of live coding and the experimentation with new technologies, human–machine agencies can be explained from an ecological perspective by taking into account different communities of practice. Our framework is designed within the live coding practice and with the focus on the figure of the developer/artist interacting with a live coding system. In this sense, the socio-political context is out of the scope of this article, we refer the reader to Blackwell et al. (2022, pp. 229–243) for a more detailed account.

2.4. Human–data interaction

Human–data interaction (HDI) is an emerging area of study that investigates how people interact with data.

Victorelli et al. (2020) conducted a literature review and proposed a set of recommendations for HDI information systems. As part of the fundamental concepts of HDI, three dimensions are presented for interacting meaningfully with data inspired by Mortier et al. (2014): *legibility*; *agency*; and *negotiability*.

Several of the surveyed publications link HDI with HCI and interaction design, and the importance of taking a human-centred perspective to avoid undesired consequences of data usage. This supports keeping the human in the loop as promoted in IML (Amershi et al., 2014; Dudley & Kristensson, 2018; Fails & Olsen Jr, 2003).

Human Data Interaction (HDI): Legibility, Agency, Negotiability (https://hdi-network.org) was a large 3-year project (2018–2021) funded by the UK Engineering and Physical Sciences Research Council (EPSRC) to foster the emerging research field of HDI and build a community of HDI researchers. The project brought together a wide range of participants and perspectives that explored the three dimensions applied to a range of areas including art, music and culture; public trust; smart cities; mental health; Internet of Things; social justice; law; ethics; and surveillance, among others. The three dimensions are defined from a practical and human-centred perspective, which is useful for artistic practices:[1]

- *Legibility* – Making the processes of sharing data about a person, and others' analysis and use of that data, comprehensible to that person.
- *Agency* – Giving a person the capacity to interact with their systems so as to control and correct the above-mentioned processes.
- *Negotiability* – Giving a person the capacity to interact with the people who do the above-mentioned analysis and use, so as to change and correct what those people do.

Data-driven AI is also permeating in live coding systems. Hence, these categories can be useful for the understanding of human–machine agencies in live coding. The three dimensions of HDI are particularly appropriate for the understanding of live-coding systems because they promote the human understanding of the hidden processes including data as an entity. This is relevant to the live coder and the audience, as we propose in Section 3.

3. Proposed framework

Inspired by the surveyed literature review, we propose a set of dimensions for understanding and evaluating human–machine agencies in live coding music performance. The starting point is the three dimensions used in the HDI literature, yet excluding *agency* because this is the term we try to define with the other dimensions. As we present the dimensions, we justify the renaming and addition of other relevant dimensions.

Legibility is stressed by the TOPLAP Manifesto:[2] 'Obscurantism is dangerous. Show us your screens.' Showing the screen makes the live coding activity of the human agent legible to the audience, but what is the machine doing, and what is the role of the data? Human–machine agency requires that the legibility of the performance applies to all agents. Legibility of machine agency can be enhanced through visualisation of code, often including agent states (McLean et al., 2010; Roberts, 2018).

Another factor that impacts legibility is the use of abstraction. Abstraction is routinely used in software development to generalise implementation details and avoid repetition. Some live coding systems make significant use of abstraction (Bell, 2013). While a high level of abstraction may result on code that is easier to understand for the audience, it also hides a significant amount of detail. Thus, finding the right level is a design decision that in live coding has different implications than in traditional software development. Specifically in the use of ML, Knotts and Paz (2021) noted the danger that the common use of black box algorithms in ML may have an impact on legibility.

Negotiability in HDI concerns the relationship with data and algorithms, and the capacity of changing them over time. When analysing NIMEs, Jourdan and Caramiaux (2023) discuss the interaction between the human and the system in terms of which parameters can be modified, how the system can be trained by the user, and how the user can intervene with the system. In live coding, Paz and Knotts (2022) look at what are the parts of the systems that are modified and how this can creatively benefit the live coder. Hence, we believe that the term *Modifiability* describes more accurately the interactions between the live coder and the algorithmic agents, and among the agents themselves, so we will use this term. When using AI and ML, a particularly relevant way to modify an agent is through learning, as noted by Tatar and Pasquier (2019) and Xambó (2022).

Predictability, or lack thereof, is an important aspect with respect to machine agency. Unlike the patterns of a sequencer or a drum machine, an agent behaving in an unpredictable manner will give a stronger impression of having a distinct agency with respect to a human performer. This was already noted in early interactive music systems research by Chadabe (1984), and it is utilised as

[1] https://hdi-network.org/about/.

[2] https://toplap.org/wiki/ManifestoDraft.

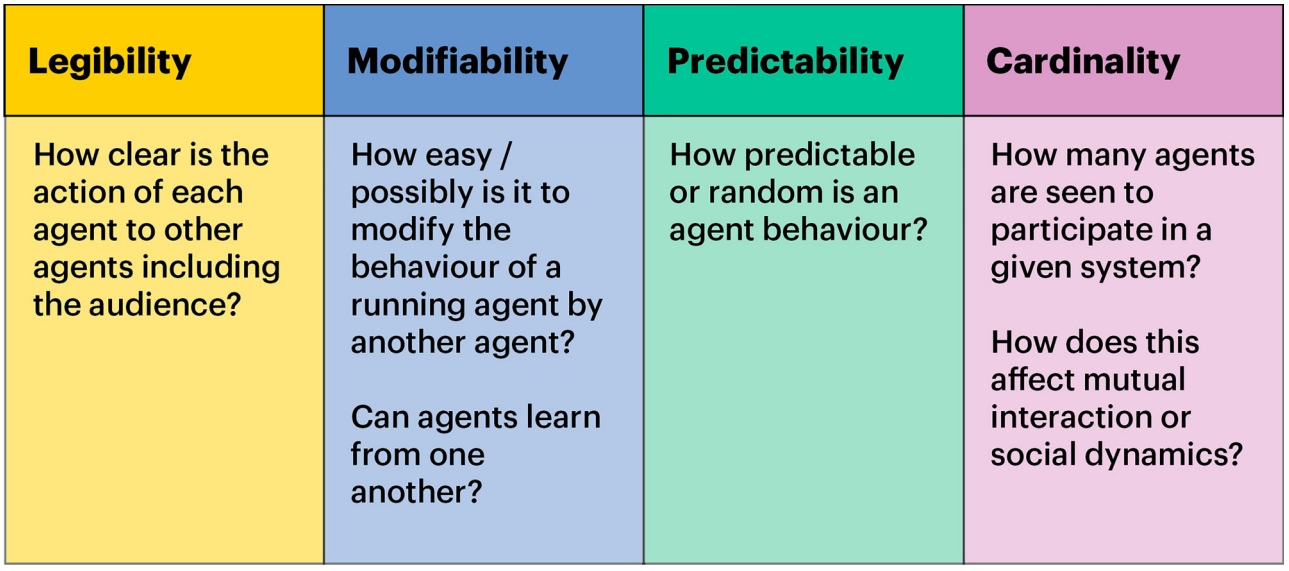

Figure 1. Proposed framework of human–machine agencies in live coding. The selected categories are: *legibility*, *modifiability*, *predictability* and *cardinality*.

a creative tool in AI-powered NIMEs (Erdem & Jensenius, 2020; Tahiroğlu et al., 2020). While a purely random behaviour may be unpredictable, this is typically not good enough for creative systems. Boden (2004) proposed that creative artefacts include an element of *novelty* and an element of *relevance* in a given context. Pure randomness may result in novelty that is not relevant, or in this context, not musically interesting. On the other hand, many ML algorithms are typically based on optimisation of a well-defined goal, which would lead to a fully predictable agency. Systems thus typically try to find a balance between pure randomness and total predictability, often seeking to facilitate serendipitous discovery of relevant or musically interesting results.

Cardinality generally describes the number of agents involved in a performance. This may refer to multi-agent systems from the computational point of view, multi-user systems from the human point of view, or any combination of the two. Cardinality was described as *social interactivity* by Xambó (2022).

To summarise, as shown in Figure 1, the proposed framework has four dimensions aiming to help understand the role of the different layers that configure human–machine agencies in live coding.

4. Case studies

In this section, we present perspectives from performing with two self-made live-coding systems developed by the authors and discuss them using the proposed framework. We adopt a practice-based approach of reflecting on the four dimensions of the proposed framework by analysing one recorded performance for each system. The first live-coding system, MIRLCa, has been developed by the first author, whereas the second live-coding system, Mob, has been developed by the second author.

Both live-coding systems have the following commonalities: (1) use of ML for training from data: yet MIRLCa is based on supervised learning whilst Mob is based on unsupervised learning. (2) usage of relatively large amounts of data: Mob is typically used with hundreds of personal sounds, whereas MIRLCa accesses more than 650,000 crowdsourced sounds from Freesound (Font et al., 2013) at the time of this writing. (3) use of ML algorithms provided by the Fluid Corpus Manipulation (FluCoMa) toolkit (Tremblay et al., 2021).

4.1. Case study 1: MIRLCa

4.1.1. Introduction

MIRLCa (Xambó, 2023) is a user-friendly live-coding environment built upon SuperCollider and the Freesound quark[3] to query Creative Commons sounds from the Freesound online database by applying music information retrieval techniques. The aim is to offer a flexible and tailorable live-coding sound-based music environment. Sound-based music has been identified as an inclusive approach to making music with a lower barrier of entry than traditional note-based music (Landy, 2011). MIRLCa uses supervised ML algorithms provided by FluCoMa to learn and predict the type of sounds that the live coder prefers to be retrieved from Freesound. Through the use of MIRLCa it is possible to 'tame'

[3] http://github.com/g-roma/Freesound.sc.

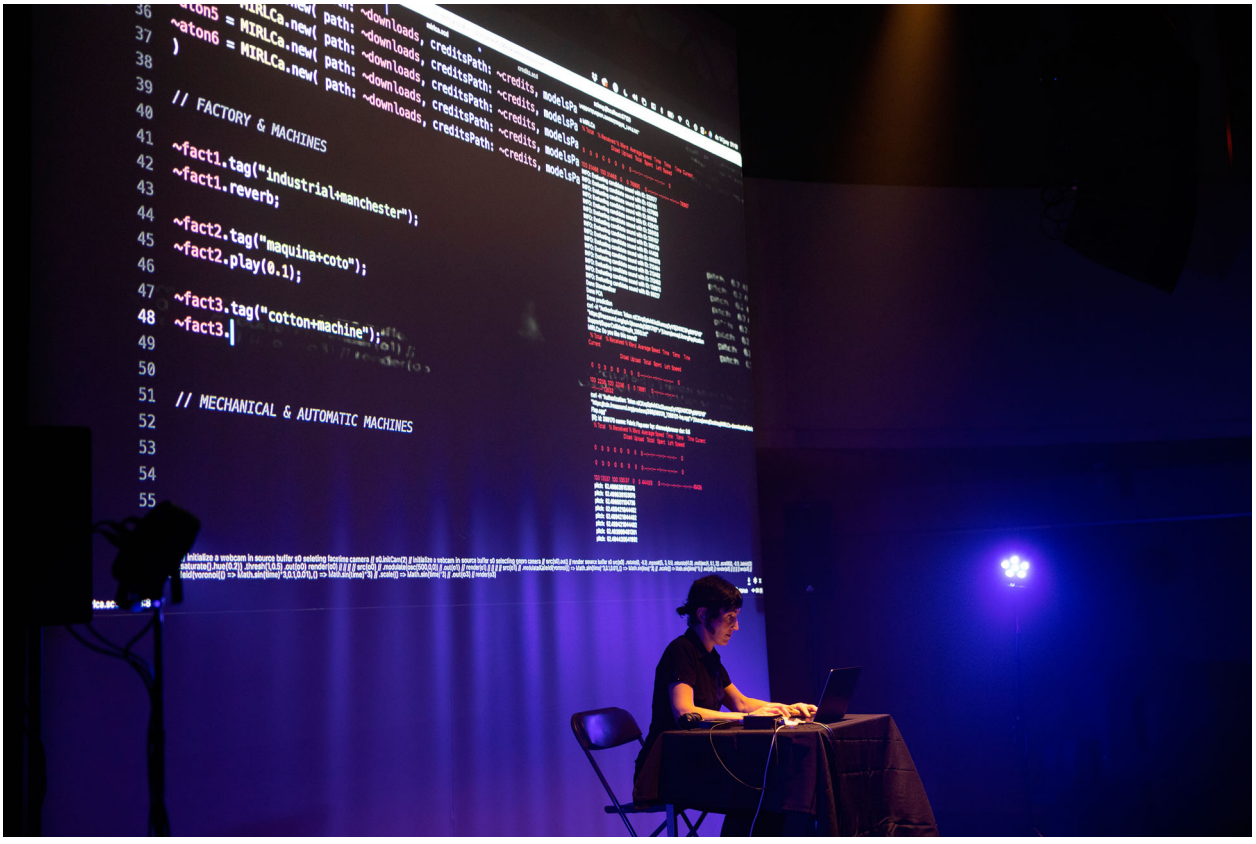

Figure 2. The first author live coding with MIRLCa at the +RAIN Film Fest, 14 June 2023, Barcelona.

the heterogeneous nature of sounds from crowdsourced libraries towards the live coder's needs. This process is enhanced with the algorithmic music possibilities brought about by live coding.

The performance analysed is *Ceci n'est pas une usine* (*This is not a factory*),[4] presented at the +RAIN Film Fest on 14 June 2023 at Sala Aranyó, Universitat Pompeu Fabra, Barcelona (Figure 2). Inspired by the location of the event at Ca l'Aranyó, a cotton factory from the 19th century that became part of a university campus in the 2000s, as well as by the current ubiquitous AI turn, this session sonically inspects the technological transformations of the site by transitioning from mechanical to automatic, synthesised and imagined soundscapes through live coding. Machine listening and ML algorithms are used through the self-built SuperCollider extensions MIRLCa and MIRLCRew2, combining sounds from the Freesound database with personal sounds in a sound-based music style. The Hydra[5] library is used for the visuals that overlap with the code using the Atom code editor. For the reflective analysis, a rehearsal video has been also used for a closer look at the screen events.[6]

[4] https://youtu.be/lQHcSbkJK5k.
[5] https://hydra.ojack.xyz.
[6] https://vimeo.com/838940013.

4.1.2. Legibility

In MIRLCa, the code is exposed with the intention of revealing an improvisational process that is not always controlled by the human live coder. The performance uses this system to create a polyphony of crowdsourced sounds from the cloud, which craft a site-specific collective story in continuous mutation.

The interface of MIRLCa shows a screen divided into two windows: the screen on the left is the code editor where the live coder writes and executes the code that generates the sound. The screen on the right is the console that provides feedback to the live coder about the status of the different processes as well as any errors that might arise. The live coder's screen is projected, hence the audience sees what the live coder sees.

The console on the right is used to give feedback about what the computational agent is doing and prompts some sentences that afford a conversation with the human live coder. The information shown in the console is relevant when making future decisions. This includes contextual information about the sound, the pool of sounds that are considered before predicting which one is the next selected sound, and even enquiries to the human live coder once the tasks are finished e.g. 'Do you like this sound?'. There is a contrast between the

high level of abstraction of the MIRLCa library functions and the amount of detailed information shown in the console.

The combination of the high-level syntax on the left with the contextual information on the right should complement and spark the imagination of both the live coder and the audience. For example, when typing `~fact1.tag("industrial + manchester")`, the selection process of the sound is shown in the console. A set of 15 sounds is selected as potential candidates related to 'industrial' and 'manchester' (e.g. sounds from Manchester and with an industrial component) and the first candidate in the list predicted as a 'good' sound is served.

The contextual information of the sound is provided, such as the Freesound identifier number, file name, author name, and duration. This information, combined with the MIRLCa syntax and the sound material, may help the audience to build an imagined narrative of the history of the sounds beyond the pure technicality of downloading a sound. The acknowledgement of the authorship of the crowdsourced sounds, and the way of selecting the best candidate, contribute to the data legibility of the performance. This also leads to an understanding of the performer as a facilitator of an experience that is built in real time from a multiplicity of voices.

Reading the code may not always be engaging for the audience, hence the use of abstract dynamic visuals that overlap the code, which can enhance the visual experience. The image source is the webcam of the laptop, which is modified with visual effects and mapped to the fast Fourier transform (FFT) of the sound for a subtle real-time interaction. The visual style is inspired by the work of Whitney (1980). The performance has four thematic sections, which are indicated by visual thematic changes combined with sonic transitions. The visuals slowly overtake the code in terms of visibility to give some intensity towards the end of the piece.

4.1.3. Modifiability

Based on a training process, the MIRLCa system aims to learn and predict the type of sounds that the live coder prefers to be retrieved from the Freesound database. A typical workflow is to train at least one model and then perform with the resulting model(s). In the analysed performance, a model of a binary classifier has been already trained for practical reasons. As such, the model is pre-defined and not modifiable. However, how the model behaves in the performance space (e.g. what sounds from the database are retrieved) influences the live coder's decisions. The MIRLCa system is mostly task-oriented: the live coder creates tasks to be pursued by the machine based on the rate at which they are completed. This gives a sense of conversation and mutual modifiability between the human live coder and the machine.

MIRLCRew2 is a complementary extension that offers a set of automatic functions over the selected sounds by MIRLCa. For example, the function `playauto` plays sounds at randomly assigned sample rates for a certain period of time and the function `playautodown` plays sounds at decreasingly sample rates until reaching stillness. This type of functions allow the live coder to modify the sounds by giving some autonomous and less predictable behaviour to specific groups of sounds. Other ways of modifying the sounds are applying a range of audio effects such as delay, reverb, vibrato, ring modulation, or low/band/high pass filters, among others.

Although not explored in the analysed performance, the MIRLCa system allows for training new models using a live-coding approach. Hence, the model could be modified during a performance by training a new model or adding more data points to the dataset. The use of training during a MIRLCa performance has been explored by Luka Frelih (Xambó, 2023, p. 286). This process involved the audience in deciding what can be considered a good or bad sound for an *algorave* (algorithmic rave) music style. This process was open to the audience, and the audience's opinion modified the training model by voting on the quality of the sounds. Once the model obtained a decent training accuracy, it was evaluated in the performance space by asking the audience to dance only if the music produced was danceable.

4.1.4. Predictability

Unless the live coder types the identification number of the sound from the Freesound database, with the currently implemented MIRLCa functions, there is no control over the sound or set of sounds that will be retrieved. Hence, the live coder moves within a sonic space (e.g. based on a tag that describes the sound, or a sound descriptor) but does not know ultimately what sound(s) will be retrieved. This lack of predictability can be also explicit through the MIRLCa's `random` function.

The analysed performance has four sonic spaces indicated as code comments: `factory & machines`, `mechanical & automatic machines`, `campus & IT`, and `dissonance & atonal`. A common pattern is to start requesting a sound or group of sounds based on tags, to then request other sounds based on similarity (e.g. `~fact3.similar`) or dissonance (e.g. `~aton1.dis`). The function `random` retrieves a random sound from the Freesound database. The use of this function was observed in other live coders (Xambó, 2023,

p. 285). This function contributes to the uncertainty and uniqueness of the performance because the likeliness of two performances using the same random sound is small. Embracing a lack of control or unpredictable results over the machine agency is found to be characteristic of AI-empowered music systems, which is well suited to improvisational practices such as live coding.

The MIRLCa system has been designed to promote serendipity instead of randomness through the inclusion of AI-driven decision-making mechanisms. Yet, sometimes, the human live coder may wish to have more control over the machine agency. In the analysed performance, personal sounds were combined with the crowdsourced sounds in order to moderate the unpredictability of the system. This strategy was found to be a promising approach in our previous research on this topic (Xambó et al., 2018).

4.1.5. Cardinality

With MIRLCa, it is possible to instantiate the same model several times as well as instantiate different models with e.g. different musical roles. Each instance can be seen as an agent. In this performance, there were around 24 instances using the same model. On average, six of them were typically playing at the same time, with transitions between them. In MIRLCa, the sounds are not synchronised, which promotes polyrhythmic structures emerging from the combination of the sounds. This contributes to the perception of the cardinality of the agents.

Four live coders associated with TOPLAP Barcelona (Ramon Casamajó, Iván Paz, Chigüire, and Roger Pibernat) used MIRLCa 'from scratch' (Villaseñor-Ramírez & Paz, 2020), adapting the library to their particular approaches and aesthetics (Xambó, 2023, p. 286). After four solo performances, the concert ended with a group improvisation 'from scratch' by the four performers, each using their models and two pairs of speakers to create a multichannel experience. Here a one-to-one mapping was explored, where each human live coder was in charge of a model with its independent behaviour, and the sum of the eight actors configured a balanced combination of human–machine agencies with four tandems of particular music styles. Regardless of whether the speakers are analysed as agents, using multi-speaker setups can be used to enhance the perception of the cardinality depending on the mapping.

4.2. Case study 2: Mob

4.2.1. Introduction

Mob is a music live coding system based on a set of agents that navigate a data *terrain*. The system was presented as an implementation of agent-based music live coding (ABMLC) (Roma, 2023). The terrain may be designed using arbitrary data, although in practice it is typically created as a 2D visualisation of a sound collection using FluidCorpusMap (Roma et al., 2021). This library uses the uniform manifold approximation and projection (UMAP) ML algorithm (McInnes et al., 2018), along with a grid layout approximation, to create a visual representation of a sample collection. In Mob, multiple live-coded agents navigate the terrain following coded functions of time and location, which are improvised by the live coder. The agents are mapped to different synthesisers that produce sound based on their location in the terrain. The coding environment is complemented with a MIDI controller that allows changing the volume of each agent.

stir bugs is a performance based on this system, which was presented remotely at the NIME 2023 conference.[7] The system was fed with a corpus of experimental recordings, created with a sound processing-oriented hardware modular set-up, using a combination of inputs and feedback. The performance explored the combination of repetitiveness and randomness afforded by the function-based control of agents. Rehearsals allowed learning the sound space and discovering interesting musical gestures. A screenshot of the system is shown in Figure 3.

4.2.2. Legibility

As a live coding environment, Mob is designed to be projected. The code is visible to the performer and audience as it is typed. While it is common in pattern-oriented live coding to use samples, the sample collection is often not visible in many systems. In contrast, in Mob the sample collection is the centre of the stage, which affords texture and timbre-based explorations, although patterns are also possible. This in turn reduces the amount of screen space available for code, so tabs are used for each agent, and only the code for the current active tab is visible. This shows that the pursuit of legibility may introduce a struggle for screen real state.

Another prominent feature is the visualisation of agents as shapes moving around the terrain. This allows performer and audience to understand what each agent is doing, mostly in terms of gesture, which results in changes to the sound. The sound synthesis function itself (typically some form of granulator or sample player) is not visible. While previous versions of this system allowed live coding of the synthesis algorithm along with the agent trajectories, in the current version the synthesisers are created in advance, and the code is not

[7] https://vimeo.com/1023077269.

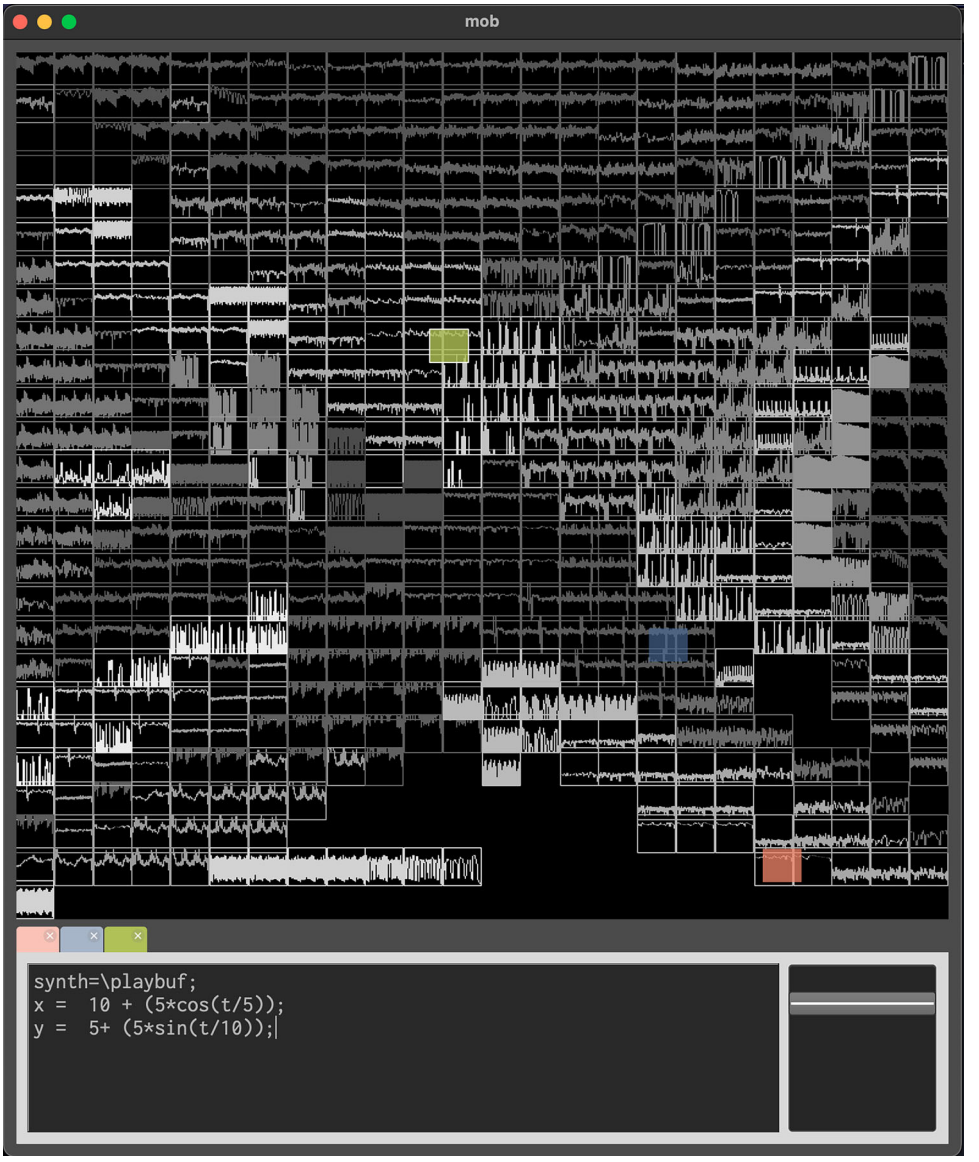

Figure 3. Screenshot of the Mob interface.

shown. This compromise benefits a focus on patterns and gestures during the performance.

With respect to the code that defines agent trajectories, *stir bugs* explored mathematical functions of time and space such as random walks, circles, or patterns, as defined in Roma (2023). This led to a certain amount of repetition in the code, which in future work may be abstracted into higher-level functions. Given that the visualisation supports the legibility of agent trajectories, a simplification of the code would probably be beneficial for the legibility of Mob performances.

4.2.3. Modifiability

The main activity in Mob is the modification of agent functions. Each agent has an autonomous behaviour, but can be re-programmed at any time. This typically leads to small tweaks once the initial idea has been coded. Another common pattern is copying and pasting the agent function from another agent, which is then modified in subtle ways. This allows creating groups of similar behaviours, but which may be operating at different parts of the terrain, or at different speeds.

There is an optimisation process in the dimensionality reduction that leads to the creation of the terrain. This happens only at initialisation time. Thus, the terrain, along with the synthesis functions, are not modifiable during the performance: they define the invariant sonic characteristics of the performance. With respect to modifiability, *Mob* is similar to other live coding environments. It is easy for the human agent to modify artificial agents, but there is no functionality for artificial agents to modify or learn from each other. Future work will explore

live coding of interactions between agents, and between agents and the terrain.

4.2.4. Predictability

The idea of controlling multiple agents quickly leads to very unpredictable behaviour. The most basic function, setting agents to walk in straight lines, typically creates loops of different lengths, without quantisation or connection to a common time measure, in the spirit of early generative music (Eno, 1996). The performer may choose to make these patterns more predictable (e.g. by making them purely horizontal or vertical, or all on the same diagonal, or on harmonic diagonals), but in general the lack of reference favours unpredictable structures. From there, it is also up to the performer to inject more or less randomness in the agent functions. For example, random walks are created by using SuperCollider random functions such as `rand` or `choose`. In all, the collective behaviour is most often impossible to predict, which may also lead to undesirable results. For this reason, a physical control of agent volume is used to moderate the global result. This shows that, like in the case of MIRLCa and previous work, unpredictability may introduce a tension between artificial and human agents.

Another interesting point, with respect to predictability, concerns the terrain generation. Like many ML algorithms, UMAP uses stochastic optimisation, but attempts to find an optimal solution. And, like other nonlinear dimensionality reduction algorithms, the solution is mostly defined *locally*: sounds that are similar should be close together in the final layout, but there is no real control of the overall distribution. For example, a map that is exactly reversed horizontally or vertically would be just as good from the optimisation point of view. Thus, the terrain generation from a given sample collection typically results in a space that can be learnt by the human over time (the sounds are the same, the groupings are the same, and they can be visually identified) but is not necessarily exactly the same every time. This can be seen as a special case of unpredictability, which happens at initialisation time and requires the human performer to adapt to the terrain.

In the agent functions, the space is addressed by spatial coordinates, which means the performer can quickly learn an approximation of what sounds are available in a given area, but rarely an exact mapping of numbers to sounds. In *stir bugs*, repetitive patterns were coded based on spatial coordinates in order to introduce highly predictive behaviour that would contrast with the randomness of some of the artificial agents.

In summary, the features in *Mob* were designed to favour unpredictable behaviours by default, which is perceived as strong machine agency, but also to make it possible to create predictable systems. This requires the human agent to learn, react and adapt, as one would do when improvising with other humans, except that artificial agents can be directly modified or silenced.

4.2.5. Cardinality

With respect to the number of agents, *Mob* can be seen as a multi-agent system inspired by agent-based models and A-life (Whitelaw, 2004), but unlike many of these models, agents are not spawned. Rather, each agent is directly supervised by the human performer when possible, perhaps more similar to livestock herding than observation of an ant colony. The use of hardware control means that there is a hard limit (e.g. eight channels) on the number of agents. When used for solo performance, the system can then be characterised by a fixed cardinality of artificial agents, which do not interact between themselves, and a human agent, which leads to a dynamic of individual supervision.

5. Discussion

The two case studies have shown the application of the proposed framework for the design, evaluation, and analysis of data-driven AI-enhanced interactive music systems for live coding. In this section, we reflect on the commonalities among the two case studies across the four dimensions and how the framework relates to ANT and other theoretical insights in the literature. We conclude by highlighting how the framework has channelled the role of data agency as a key actor in AI-enhanced live coding.

Following ANT, live coding performances can be described as networks of agents that embed multiple human and machine agencies. A common technique of ANT is *blackboxing* (Callon, 1986), where part of the network is simplified and viewed as an agent. Software environments commonly used in live coding can be seen as a black box: they are another actor in the network of a live coding performance, but they also contain a network of actors, including data and artificial agents. While the proposed framework can be applied to both levels, the two case studies have focused mostly on the software environment box. In live coding performances, the code and the interface of the coding environment are typically shown to the audience, and also strongly influence the actions of the performer. Thus, the analysis of the relationships between code, data and user interface (the black box) based on the framework also supports the understanding of the relationships between the actors in the live coding performance network, including audience, performance, laptop, and even the speaker set-up.

We have stressed *legibility* as a dimension that connects live coding with HDI. Legibility is the main mechanism for connecting audience, performers and artificial agents. Beyond the code and the performer actions, data, algorithms and artificial agent states are of interest for legibility of machine agency. While the use of ML typically requires abstractions that hide the detail of the mathematical operations involved in training and inference, both case studies have shown that the use of ML in live coding offers new opportunities for improving legibility with respect to the use of data and the conversations between agents.

Modifiability has been described mostly in terms of connection between the human performer and autonomous agents. Although none of the two case studies described modifications between artificial agents, this is a common feature in interactive music and agent-based systems that can be easily incorporated in live coding. In this sense, live coding of agents that learn from one another, or interact with each other, is a promising research direction in music performance.

The lack of *predictability* appeared to bring a distinctive aesthetic value to the use of agents in music performance. This can be an interesting space of exploration given the numerous possibilities that it can entail, which can be combined with more controlled behaviours.

Finally, we have described *cardinality* both in terms of multiple artificial agents and multiple human agents. There are of course many potential configurations involving different kinds of actors, and in some cases, the exact number may be irrelevant or not fully known. However, this dimension allows explaining the multiplicity of actors, especially with respect to traditional accounts (e.g. 'a human and a machine').

In all, the proposed framework generally supports the view of live coding as an entanglement (Morrison & McPherson, 2024) of human and non-human actors influencing each other to create a specific performance, where often overseen actors, such as data or the audience, play also a key role and are part of a complex network of agencies.

It is perhaps worth pointing at *data agency* as a concept of special relevance for AI-powered live coding, which was reflected in our case studies: these systems often deal with large amounts of data that would be too time-consuming for humans to tackle manually. Moreover, any selected data brings its own bias and character. With the adoption of DL, datasets will likely continue to grow, which suggests the importance of endorsing clear principles of data transparency and acknowledgement of authorship as shown in our case studies. Bryan-Kinns (2024) suggests finding artistic representations of the complexities behind DL models, termed XAIxArts, which could also be applied to live coding. Data bias is of big concern to avoid the promotion of inequality biases such as racism or sexism (D'Ignazio & Klein, 2023; Jourdan & Caramiaux, 2023). Yet, for artistic performance, small datasets are commonly generated by individuals, and their deliberate biases are often a signature of the musical piece (Murray-Browne & Tigas, 2021).

6. Conclusion

In this article, we investigated human–machine agencies in live coding music performance. We proposed a theoretical framework of four dimensions informed by the literature on interactive music systems, NIMEs, live coding and HDI. The four dimensions tackle agency from different angles: *legibility*, *modifiability*, *predictability* and *cardinality*. We analysed two self-built live-coding systems using the four dimensions from a practice-based perspective and discussed how agency relates to ANT, entanglement HCI, and the role of data.

Here, we focused our observations on live-coding systems powered by traditional ML. We expect similar observations can be made when using DL, but a formal study should be conducted. In future work, we hope to also study the use of generative DL systems in the context of live coding. This will likely bring new qualitative agency dimensions when generating audio or using generative language models.

This article has focused on human–machine agencies specifically in live coding. Our findings can be applied to other domains, such as interactive music beyond live coding, or other forms of interactive computational art. We expect that network-based analysis of artistic performance will continue to be key to the understanding of human and non-human agencies in data-driven AI-powered interactive systems.

Acknowledgments

The first author would like to thank Koray Tahiroğlu for the invitation to present the seminal work of this article in Helsinki, Finland, in November 2022, with the invited talk "HCI meets AI in Live Coding: A Practitioner's Perspective" at the *Symposium Technoscientific Practices of Music; New Technologies, Instruments and Agents*, and the vibrant conversations emerged in the symposium. The authors thank the reviewers for their insightful comments and the editors for the opportunity to present this work.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

The project *MIRLCAuto: A Virtual Agent for Music Information Retrieval in Live Coding* was funded by the EPSRC HDI Network Plus Grant (EP/R045178/1). This work was partially upported by the Academy of Finland Research Fellow Project Digital Musical Interactions - Instruments - Performances under Grant 316549.

References

Agres, K., Forth, J., & Wiggins, G. A. (2016). Evaluation of musical creativity and musical metacreation systems. *Computers in Entertainment*, *14*(3), 1–33. https://doi.org/10.1145/2967506

Amershi, S., Cakmak, M., Knox, W. B., & Kulesza, T. (2014). Power to the people: The role of humans in interactive machine learning. *AI Magazine*, *35*(4), 105–120. https://doi.org/10.1609/aimag.v35i4.2513

Arias, J., Desainte-Catherine, M., & Dubnov, S. (2016). Automatic construction of interactive machine improvisation scenarios from audio recordings. In P. Pasquier, O. Bown, & A. Eigenfeldt (Eds.), *Proceedings of the 4th International Workshop on Musical Metacreation (MUME 2016)*. https://musicalmetacreation.org/proceedings/mume-2016/

Bell, R. (2013). An approach to live algorithmic composition using conductive. In iOhannes m zmölnig & Peter Plessas (Eds.), *Linux Audio Conference 2013 Proceedings* (pp. 29–36). Institute of Electronic Music and Acoustics, University of Music and Performing Arts Graz.

Biles, J. (1994). GenJam: A genetic algorithm for generating jazz solos. In *Proceedings of the International Computer Music Conference* (pp. 131–137). Michigan Publishing.

Blackwell, A. F., Cocker, E., Cox, G., McLean, A., & Magnusson, T. (2022). *Live coding: A user's manual*. MIT Press.

Boden, M. A. (2004). *The creative mind: Myths and mechanisms*. Routledge.

Brown, A. R. (2016). Understanding musical practices as agency networks. In F. Pachet, A. Cardoso, V. Corruble, & F. Ghedini (Eds.), *Proceedings of the Seventh International Conference on Computational Creativity (ICCC 2016)*. Sony CSL Paris.

Bryan-Kinns, N. (2024). Reflections on explainable AI for the arts (XAIxArts). *Interactions*, *31*(1), 43–47. https://doi.org/10.1145/3636457

Bullock, J., & Momeni, A. (2015). ml.lib: Robust, cross-platform, open-source machine learning for Max and Pure Data. In E. Berdahl, & J. Allison (Eds.), *Proceedings of the International Conference on New Interfaces for Musical Expression* (pp. 265–270). Louisiana State University.

Callon, M. (1986). The sociology of an actor-network: The case of the electric vehicle. In M. Callon, J. Law, & A. Rip (Eds.), *Mapping the dynamics of science and technology: Sociology of science in the real world* (pp. 19–34). Springer.

Chadabe, J. (1984). Interactive composing: An overview. *Computer Music Journal*, *8*(1), 22–27. https://doi.org/10.2307/3679894

Dahlstedt, P. (2021). Musicking with algorithms: Thoughts on artificial intelligence, creativity, and agency. In E. R. Miranda (Ed.), *Handbook of artificial intelligence for music* (pp. 873–914). Springer International Publishing.

Déguernel, K., Vincent, E., Nika, J., Assayag, G., & Smaïli, K (2019). Learning of hierarchical temporal structures for guided improvisation. *Computer Music Journal*, *43*(2-3), 109–124. https://doi.org/10.1162/comj_a_00521

DeSmith, M. O., Piepenbrink, A., & Kapur, A. (2020). SQUISHBOI: A multidimensional controller for complex musical interactions using machine learning. In R. Michon, & F. Schroeder (Eds.), *Proceedings of the International Conference on New Interfaces for Musical Expression* (pp. 353–356). Birmingham City University.

D'Ignazio, C., & Klein, L. F. (2023). *Data feminism*. MIT Press.

Dudley, J. J., & Kristensson, P. O. (2018). A review of user interface design for interactive machine learning. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, *8*(2), 1–37. https://doi.org/10.1145/3185517

Eno, B. (1996). Generative music: Evolving metaphors, in my opinion, is what artists do. *In Motion Magazine*, *7*(2). https://inmotionmagazine.com/eno1.html

Erdem, C., & Jensenius, A. R. (2020). RAW: Exploring control structures for muscle-based interaction in collective improvisation. In R. Michon, & F. Schroeder (Eds.), *New Interfaces for Musical Expression, NIME 2020* (pp. 477–482). Birmingham City University.

Fails, J. A., & Olsen Jr, D. R. (2003). Interactive machine learning. In *Proceedings of the 8th International Conference on Intelligent User Interfaces* (pp. 39–45). Association for Computing Machinery.

Fiebrink, R., & Caramiaux, B. (2018). The machine learning algorithm as creative musical tool. In R. T. Dean & A. McLean (Eds.), *The Oxford handbook of algorithmic music* (pp. 181–208). Oxford University Press.

Fiebrink, R., & Sonami, L. (2020). Reflections on eight years of instrument creation with machine learning. In R. Michon, & F. Schroeder (Eds.), *Proceedings of the International Conference on New Interfaces for Musical Expression* (pp. 237–242). Birmingham City University.

Fiebrink, R., Trueman, D., & Cook, P. R. (2009). A meta-instrument for interactive, on-the-fly machine learning. In *Proceedings of the International Conference on New Interfaces for Musical Expression* (pp. 280–285). Carnegie Mellon University.

Font, F., Roma, G., & Serra, X. (2013). Freesound technical demo. In *Proceedings of the 21st ACM International Conference on Multimedia* (pp. 411–412). Association for Computing Machinery.

Jourdan, T., & Caramiaux, B. (2023). Machine learning for musical expression: A systematic literature review. In M. Ortiz & A. Marquez-Borbon (Eds.), *Proceedings of the International Conference on New Interfaces for Musical Expression* (pp. 319–331). https://nime.org/proc/nime2023_46/index.html

Knotts, S., & Paz, I. (2021). Live coding and machine learning is dangerous: Show us your algorithms. In *International Conference on Live Coding*. Zenodo.

Landy, L. (2011). Sound-based music 4 all. In *The Oxford handbook of computer music*. Oxford University Press.

Latour, B. (2005). *Reassembling the social: An introduction to actor-network-theory*. Oxford University Press.

Macionis, M. J., & Kapur, A. (2018). Sansa: A modified sansula for extended compositional techniques using machine learning. In T. M. Luke Dahl, & D. Bowman (Eds.), *Proceedings of the International Conference on New Interfaces for Musical Expression* (pp. 78–81). Virginia Tech.

McCallum, L., & Grierson, M. S. (2020). Supporting interactive machine learning approaches to building musical

instruments in the browser. In R. Michon, & F. Schroeder (Eds.), *Proceedings of the International Conference on New Interfaces for Musical Expression* (pp. 271–272). Birmingham City University.

McInnes, L., Healy, J., & Melville, J. (2018). *UMAP: Uniform manifold approximation and projection for dimension reduction*. Preprint arXiv:1802.03426.

McLean, A., Griffiths, D., Collins, N., & Wiggins, G. (2010). Visualisation of live code. In A. Seal, J. P. Bowen, & Kia Ng (Eds.), *Electronic Visualisation and the Arts (EVA 2010)* (pp. 26–30). BCS Computer Arts Society.

Morrison, L., & McPherson, A. (2024). Entangling entanglement: A diffractive dialogue on HCI and musical interactions. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (pp. 1–17). Association for Computing Machinery.

Mortier, R., Haddadi, H., Henderson, T., McAuley, D., & Crowcroft, J. (2014). Human-data interaction: The human face of the data-driven society. SSRN Electronic Journal.

Murray-Browne, T., & Tigas, P. (2021). Latent mappings: Generating open-ended expressive mappings using variational autoencoders. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. NYU Shanghai.

Næss, T. R., & Martin, C. P. (2019). A physical intelligent instrument using recurrent neural networks. In M. Queiroz, & A. Xambó Sedó (Eds.), *Proceedings of the International Conference on New Interfaces for Musical Expression* (pp. 79–82). UFRGS.

Pachet, F. (2003). The continuator: Musical interaction with style. *Journal of New Music Research*, *32*(3), 333–341. https://doi.org/10.1076/jnmr.32.3.333.16861

Paz, I., & Knotts, S. (2022). Live coding machine learning: Finding the moments of intervention in autonomous processes. *Computer Music Journal*, *46*(4), 81–96. https://doi.org/10.1162/comj_a_00663

Roads, C. (1996). *The computer music tutorial*. MIT Press.

Roberts, C. (2018). Realtime annotations & visualizations in live coding performance. In *Proceedings of the 2018 LIVE Programming Workshop*. https://liveprog.pubpub.org/

Roma, G. (2023). Agent-based music live coding: Sonic adventures in 2D. *Organised Sound*, *28*(2), 231–240. https://doi.org/10.1017/S1355771823000274

Roma, G., Xambó, A., & Freeman, J. (2018). User-independent accelerometer gesture recognition for participatory mobile music. *Journal of the Audio Engineering Society*, *66*(6), 430–438. https://doi.org/10.17743/jaes.2018.0026

Roma, G., Xambó, A., Green, O., & Tremblay, P. A. (2021). A general framework for visualization of sound collections in musical interfaces. *Applied Sciences*, *11*(24), 11926. https://doi.org/10.3390/app112411926

Rowe, R. (2004). *Machine musicianship*. MIT Press.

Rutz, H. (2016). Agency and algorithms. *Journal of Science and Technology of the Arts*, *8*, 73. https://doi.org/10.7559/citarj.v8i1.223

Smailis, D., Andreopoulou, A., & Georgaki, A. (2021). Reflecting on the musicality of machine learning based music generators in real-time jazz improvisation: A case study of OMax-ImproteK-Djazz. In *Proceedings of the 2nd Joint Conference on AI Music Creativity*. AIMC.

Suchman, L. A. (1987). *Plans and situated actions: The problem of human-machine communication*. Cambridge University Press.

Tahiroğlu, K., Kastemaa, M., & Koli, O. (2020). Al-terity: Non-rigid musical instrument with artificial intelligence applied to real-time audio synthesis. In R. Michon, & F. Schroeder (Eds.), *Proceedings of the International Conference on New Interfaces for Musical Expression* (pp. 337–342). Birmingham City University.

Tanimoto, S. L. (2013). A perspective on the evolution of live programming. In *2013 1st International Workshop on Live Programming (LIVE)* (pp. 31–34). IEEE.

Tatar, K., & Pasquier, P. (2019). Musical agents: A typology and state of the art towards musical metacreation. *Journal of New Music Research*, *48*(1), 56–105. https://doi.org/10.1080/09298215.2018.1511736

Tremblay, P. A., Roma, G., & Green, O. (2021). Enabling programmatic data mining as musicking: The fluid corpus manipulation toolkit. *Computer Music Journal*, *45*(2), 9–23. https://doi.org/10.1162/comj_a_00600

Victorelli, E. Z., Dos Reis, J. C., Hornung, H., & Prado, A. B. (2020). Understanding human-data interaction: Literature review and recommendations for design. *International Journal of Human-Computer Studies*, *134*, 13–32. https://doi.org/10.1016/j.ijhcs.2019.09.004

Villaseñor-Ramírez, H., & Paz, I. (2020). Live coding from scratch: The cases of practice in Mexico City and Barcelona. In *Proceedings of the 2020 International Conference on Live Coding*. University of Limerick.

Wang, C.-I., & Dubnov, S. (2015). The variable Markov Oracle: Algorithms for human gesture applications. *IEEE MultiMedia*, *22*(4), 52–67. https://doi.org/10.1109/MMUL.2015.76

Whitelaw, M. (2004). *Metacreation: Art and artificial life*. MIT Press.

Whitney, J. (1980). *Digital harmony*. Byte Books.

Xambó, A. (2022). *Virtual agents in live coding: A short review*. eContact! 21.1. June 30, 2024. https://econtact.ca/21_1/xambosedo_agents.html.

Xambó, A. (2023). Discovering creative commons sounds in live coding. *Organised Sound*, *28*(2), 276–289. https://doi.org/10.1017/S1355771823000262

Xambó, A., Roma, G., Lerch, A., Barthet, M., & Fazekas, G. (2018). Live repurposing of sounds: MIR explorations with personal and crowdsourced databases. In T. M. Luke Dahl, & D. Bowman (Eds.), *Proceedings of the International Conference on New Interfaces for Musical Expression* (pp. 364–369). Virginia Tech.