Collaborative Composition with Creative Systems

Arne Eigenfeldt
School for the Contemporary Arts
Simon Fraser University
Vancouver, Canada
arne_e@sfu.ca

Abstract

Computational assistants are now regularly used in creative industries, and their use in music dates back practically to the 1950s, and theoretically to the 19th century. The author describes specific instances of interactive and algorithmic music systems from the last 50 years as examples of artistic assistants, suggesting that such systems exhibit only a limited role in the creative process. True collaboration, as evidenced in traditional human practices, requires greater autonomy, independence, and potential influence; he describes his collaboration with his most recent metacreative system, *Moments.*

Keywords

Collaboration, composition, creativity, metacreation, generative music.

Introduction

The possibility of having a virtual creative assistant has intrigued artists since the mid-nineteen century, when Ada Lovelace speculated upon the musical potential of Babbage's Analytical Engine [Toole, 1996]. There are many instances of computers being used as creative assistants, helping to fulfill the vision of [usually) a single artist; within music, an entire branch of computer music has been dedicated to interactivity in performance [Garnett, 2001]. However, I argue that while a creative assistant is certainly useful, such a role is always secondary to a principal artist; collaboration with a software system, in which both contribute equally to the final artwork, is now becoming a possibility through the use of new techniques borrowed from artificial intelligence.

This paper will describe how my use of my own software has evolved from assistant to collaborator, focusing upon a recent metacreative system, *Moments*.

The State of the Art in the 1980s

I have been creating software systems to aid in my musical composition for over thirty years. In the mid-1980s, when I was still a graduate student, a new paradigm was emerging: creative artists, with a mixture of interest and effort, could learn to code, and use new affordable home computers to help in their creative process. Within my area of musical composition, the Atari ST and the Apple Macintosh computers could be programmed to control relatively inexpensive commercial synthesizers, such as the Yamaha DX-7, using a newly standardized method of communication, the Musical Instrument Digital Interface, or MIDI. Such personal systems, while clearly idiosyncratic and ad-hoc, allowed for a more mercurial approach lacking in electroacoustic music at the time. MIDI was originally intended as a performance interface, allowing musicians to couple and control inexpensive digital synthesizers; but by putting a MIDI port on the back of the affordable Atari ST computer, new possibilities for compositional control were afforded [Dorfman and Young, 1986].

One can wistfully look back at this period, where creative coders were unbound by commercial aesthetics and molds, completely free to explore their individual artistic directions. The (now) ubiquitous music sequencer, itself replicating the older model of the tape studio (e.g. Logic's, Pro Tools', and Cubase's transport and multi-track format) was yet to be released. That said, since there were no actual music programming languages, artists were forced to not only learn, but also often bend, existing programming environments to meet their unique requirements.

One paradigm, of which George Lewis' *Voyager* system is the best example, was that of an improvising ensemble which interacted with a live instrumental performer through rudimentary machine listening [Lewis, 1999]. The problem of understanding what the performer was doing was partially solved for us, in that a commercial pitch detection system that converted live monophonic audio to MIDI events – the *IVL Pitchrider* – was available. Using the *Pitchrider*, it was possible to track a live performer, and determine the relative range she was playing (i.e. low versus high notes), how active she was (i.e. how many new MIDI events were being generated every second), how loud she was playing, and whether she was playing rhythmically or not (extracting tempo and rhythmic patterns was quite a bit more complicated), and, perhaps most useful, whether she was even playing.

These systems were viewed as creative partners, real-time systems with which artists could interact in performance that could produce output that designers might not have directly dictated, but, ideally, *could have*. The surprise that designers hoped to discover

while interacting with their software was engendered through constrained random procedures [Winkler, 2001], allowing the computer to select from a predetermined set of possibilities: for example, playing pitches from a given set, or outputting note events at a specific tempo, selecting from a weighted set of rhythmic subdivisions. I doubt if anyone considered their software truly "creative" or "intelligent": we were happy if the system produced novel and somewhat unexpected output. Chadabe suggested that "indeterminacy is the heartbeat of the interactive system" [Chadabe, 1996], while I have suggested that indeterminacy is a poor substitute for intelligent musical decision-making [Eigenfeldt, 2007].

Cope and Early Generative Ideals

We can look back now and determine that the cutting-edge artificial intelligence research in music in the 1980s was following the general AI trend of the time in attempting to create expert systems: in the case of music, software that could voice Bach chorales properly [Ebcioğlu, 1988]. It was at this time that David Cope began to create his Experiments in Musical Intelligence (EMI, or Emmy), a creative system whose goal was initially to produce new music, but quickly veered into reproducing existing musical styles [Cope, 1987]. Cope produced a wonderful compendium of publications tracing his technical and aesthetic development over the course of creating Emmy, culminating in his description of his decision to essentially kill Emmy by destroying her database in 2003 [Cope, 2005].

Since Emmy used machine learning to generate her own rules, Cope's interaction went beyond only coding her; by deciding the compositions from which Emmy should learn – i.e. by curating the database, or corpus – Cope was making selective decisions that contributed to the resulting music. Furthermore, Cope's work-flow had Emmy generate dozens, if not hundreds, of generations, from which he would select what he subjectively felt was the best version. As an artist, he felt he had the right to make such selections without explaining these decisions beyond simple intuition or aesthetic preference; unfortunately, the scientific community considered such decisions "cherry-picking". Cope always claimed that "what matters most is the music" [Cope, 2005], which simply doesn't fly with scientists, who countered "with hand-coded rules of whatever kind, we can never get away from the claim that the creativity is coming from the programmer and not the program" [Wiggins, 2008]. Clearly what Wiggins objected to was not the fact that a system could generate music – Wiggins is a leader in the field of computational creativity – but that Cope suggested that Emmy was "creative" on her own. The computational creativity community believes that the final selection must be made by the system itself, and the only way a system can do that is if it understands what it is doing, and can judge its output based upon its own clear goals [Ventura, 2015]. Unfortunately, successful computational aesthetics are still a ways off [Galanter, 2012]; in the meantime, artists are still required to have some influence upon making the final selection.

Sidestepping the argument of pure machine creativity, more interesting is Cope's subsequent creation of Emmy's "daughter", Emily Howell. Whereas Emmy reproduced music within a clear style by learning the style's rules from a corpus, Emily Howell produces new music through association, involving a conversation between Cope and the system: a novel, but certainly ad hoc, method [Cope, 2013]. Since Cope no longer claims Emily Howell as being creative on her own, he has been left to his own musical devices by the scientific folks.

Generative Music & Metacreation

An alternate approach to generating music in real-time was using computers to algorithmically create traditional musical notation for later performance by acoustic instruments. Using computational means to produce fixed music had existed for several decades; indeed, the first use of a computer in music was Hiller's *Illiac Suite* of 1958, in which algorithms were created to produce number streams, which in turn were hand transcribed into musical notation and played by a string quartet [Hiller, 1964]. Other examples include Xenakis programming stochastic methods [Xenakis, 1971], and Koenig's Project II, which generated serial music [Koenig, 1978]. Barlow used the computer as a compositional assistant to produce acoustic music beginning in the 1970s [Barlow, 2010]. Many other examples exist, demonstrating what composers in the field considered algorithmic music: simply put, music produced by algorithms.

In 1996, Eno came up with the term Generative Music, which codified a practice implicit in algorithmic systems, but not explicitly stated: the notion that a system could produce multiple iterations of a work, and each would be considered viable and representative of the work itself [Eno, 1996]. Realtime interactive systems (e.g. Lewis' Voyager) produced a single instance – that heard in concert – but were capable of producing multiple alternative versions; non real-time systems could produce multiple outputs (e.g. Cope's Emmy), but composer-designers selected one version that they deemed to be the best. Koenig went so far as to state that it was inherent upon the designer to not alter the output in any way, but only alter the

underlying algorithm which produced the output [Koenig, 1983]. His goal reflected the modernist aesthetic of searching for the perfect fixed object; Eno's embodied the postmodernist acceptance of continual change.

Galanter points out that generative art, and by extension generative music, does not require computational means, offering a more inclusive definition that posits the use of autonomous systems and processes creating or contributing to the creation of an artwork [2003]. Processes have a long history in music – from the rules to generate a 14th century musical canon to 20th century serialism, for example – and haven't required software for their execution. However, recent applications of artificial intelligence, evolutionary algorithms, and cognitive science have created a contemporary approach to generative art, known as metacreation [Whitelaw, 2004]. Musical metacreation (MuMe) looks at all aspects of the creative process and their potential for systematic exploration through software [Pasquier et al., 2016].

Musebots

The MuMe community has produced a variety of creative systems with rich musical results. However, the complexity of individual projects has resulted in idiosyncratic, non-idiomatic systems, created by artist-programmers with ad hoc means. A push was made to create a shared platform for MuMe interaction that would allow multiple practitioners to communally develop individual musical agents, or musebots [Bown, et al. 2015]. Within a year of developing the musebot protocol, over five-dozen individual musebot contributions existed. These were presented in continuous running installations [Eigenfeldt, 2016], curated into ensembles of 3–7 musebots. As the musebots generated music in a variety of styles, curation was a creative activity in itself; it was found that combining diverse styles rarely produced satisfying music, while combining similar styles produced music that was surprising and musically successful (as evidenced by their successive acceptances at international festivals).

As the main musebot developer, I found myself imagining ensembles in advance, envisaging how individual musebots might interact, and coding new musebots with that goal in mind. Because musebots themselves are self-contained, and react to other musebots, conceiving of a final musical result was a process in itself: one cannot control the exact output of a musebot, only how it interprets, and reacts to, its environment. While I recognized the musical limits of the musebot ensemble [see Eigenfeldt, 2017], such creative collaboration with (moderately) intelligent systems stoked my interest in pursuing actual collaboration with more powerful musebots.

Interaction versus Collaboration

As mentioned, early practitioners of real-time systems aimed for interactivity. Chadabe [1984] describes in detail his approach, which he refers to as *design-then-do*; he would create a system with which he would later interact in performance. What becomes clear in this paradigm is that the interaction is instantaneous: his actions cause an immediate reaction in the system, which then generates musical material that is heard, and can be reacted to, which in turn causes a further reaction by the system. Such direct relationships are the basis of the longstanding notion of human-computer interaction [Card et al., 1984], which continues to be an active field of research inclusive of artistic media [1]. While the potential for artistic partnership between human and computer may nominally exist within such systems, I would argue that it is a relationship much different than a collaborative one.

In viewing human interaction in creative performance, performers may interact with one another to alter a performance as it is occurring: for example, a string quartet executing a collective slowing down in tempo, or contact dancers devising movements on the spot. However, this is fundamentally different than artists collaborating on a creation, which tends to occur *before* a performance in the design process. Collaborators influence one another's artistic planning rather than only its execution.

Within music, computer-assisted composition offers the human composer assistance in executing their artistic notions. Assayag et al. describe some powerful tools designed to offer the user suggestions, particularly on the more difficult concepts regarding structure [2006]. Such research can be considered a branch of computer-assisted creativity, which often goes as far as modeling cognitive processes in order to aid creativity [López-Ortega, 2013]. However, both examples remain creative assistants, which are fundamentally different than collaborators; the former helps a single creator achieve something new, while the latter shapes that vision by offering alternatives as well as supporting views.

I propose that collaboration *can* occur between artists and systems, but in order to attain an equality between the two – necessary for the latter to become more than assistant – the system must potentially alter the artist's vision with suggestions that are unexpected, and that these proposals occur prior to the generation

[1] See, for example, http://www.nime.org/

so as to affect structural, not just surface, aspects. Lastly, I propose that this can only be done if the system exhibits some form of creative intelligence, combined with a developed trust between the artist and the creative software.

Collaboration with Intelligent Systems

In 2016, I began to use musebots in a closed system, each one developed by myself for a specific purpose within a specific generative work, *Moments*. Rather than employing a bottom-up, self-organising, improvisation model of the musebot ensemble, *Moments* is a combination of top-down and bottom-up methodologies. A `ParamBOT` generates a structure for each ten-minute composition, consisting of individual sections within which audio musebots negotiate the musical details, given the goals provided by the `ParamBOT`. A section, or moment, is comprised of a static entity – for example, a single harmony; moments avoid development and goal-directed behaviour, although the potential for processes to provide variation in the surface design are possible. Subsequent moments are contrasting, often dramatically, with one another, as their internal organisation and concerns must be different; as a result, changes between moments result in what Kramer refers to as discontinuity [1988].

One initial goal of *Moments* was the generation of complete musical structures, a complex issue not fully solved in generative music [Eigenfeldt et al., 2016]. *Moments* creates such structures without any human interaction, and continually composes, then performs, ten-minute compositions.

A great deal of my time has been spent fine-tuning the ensembles for *Moments*. As before, ensembles are specific collections of musebots active in each composition; in the case of *Moments*, the musebots are all my own, and currently number seven unique audio generating types. Ensemble scripts also dictate (to a degree) how individual musebots should behave through setting personality parameters, such as impatience, persistence, vitality, consistency, and compliance. The act of specifying an ensemble is a creative act in itself, a curation of potential actions without dictating specifics.

Because *Moments* is completely autonomous, and no interactivity is possible, it is outside the improvisational model of Chadabe's interactive composition. Like most generative systems, *Moments* is software created through an iterative design process: continual testing of the system results in fine-tuning of its code and/or parameters: what Oliver Bown describes as "finding the sweet spot"[2]. One could therefore argue that coding such a system is in an act of collaboration, as the output of any metacreative system will always (or at least hopefully) be surprising, and perhaps suggest new directions to its coding and revision. However, the collaboration is only in one direction: while the design and test methodology may result in alterations to the code due to successive listening, the resulting changes to the code are permanent, thus ending that specific avenue of collaboration.

I have found that my own relationship with *Moments* is different than with any other generative system that I have created [see, for example, Eigenfeldt, 2009]. Because it lacks interactivity, I am required to listen to the complete compositions that it produces; rather than immediately altering surface features as I hear them, I listen to its overall evolution, and react to its musical structure, perhaps similar to how a director might take notes of a complete run of a show, rather than stopping to fix problems in individual scenes. Like human improvisers, musebots are independent and autonomous; I can suggest and possibly provoke, but, unlike chamber musicians reading fixed scores, I cannot force them to perform specific actions.

Moments is a complex system. Like any such system, it is difficult to understand the complex interactions that occur while it is operating, even as its designer. As already mentioned, parametric adjustments – such as raising an individual musebot's volume at a certain point – are not possible, since that parameter was calculated for that instant due to many underlying factors. Instead of interacting with the system directly to adjust the volume, it is necessary to understand *why* the musebot may be playing at a low volume: does it's personality have too low a vitality attribute so as not to be able to sustain longer periods of higher volume? A higher vitality attribute may solve this, but it will also influence how the musebot behaves in other sections. Was the overall request structural request for low volume being met by this agent, thus demonstrating a high compliance, while other agents were non-compliant and playing louder than requested? Matching all musebot compliance levels more closely may result in a more uniform volume level, but it will also disrupt the overall variety between them. This type of collaboration between musebots and myself has led to not only refining specific ensembles, but also suggesting new ones.

Similarly, extended listening has suggested alternative versions of *Moments* that create distinctly

[2] Personal communication

different artworks[3]; *Moments: Polychromatic* uses a variety of audio producing methods that highlight their timbral differences; *Moments: Monochromatic* uses a single synthesis technique that emphasizes timbral similarity opposed to a live performer's input. In each case, I have created related musebots – cousins, if you will – that are placed in completely new environments in an effort to discover new relationships between them, as well as new relationships between them and myself.

Perhaps most surprisingly to me is that *Moments* is attempting to defy its own purpose: each composition and performance was meant to be considered a unique entity that exists only once in a mercurial form, a musical quantum that cannot be revisited. But due to the sophistication of the work that it produces, often generating works of beauty that have gone beyond what I could independently conceive, I feel the need to record its outputs for posterity. My intention may be momentary, but as collaboration, its music deserves to be preserved. The system is clearly complex, and I consider it complete, in the sense that it has been producing successful music presented at international festivals for almost a year. Like many metacreative systems, its output cannot be predicted; however, my relationship with it is not one in which I simply "let it run" on its own; instead, I find that its musical decisions continually provoke me to developing new ensembles, and even new musebots in a way that a theatrical director might decide to switch actors, or a musical director might decide to alter the makeup of her ensemble; in these latter examples, the decisions can only occur when one trusts the other performer/creators to produce something new and unexpected, in the same way that I trust *Moments* to produce a successful composition, even if I add a new musebot into the ensemble.

Conclusion

In recent months, there has been a great deal of media interest in the notion of art created by artificial intelligence[4], with a somewhat predictable reaction and fear of artists being replaced by software. This is reminiscent of any occasion in which new technology has been introduced within society, followed by a propagated fear that the current status quo would be disrupted in a negative way. Artificial intelligence will no doubt change the way artists create, if it hasn't already done so. As described in this paper, many artists in the past decades have already attempted to create a symbiotic relationship with technology; to interact with it is such a way that it assists them in their creative process. Making the tools more powerful by coding autonomous systems that might produce creative ideas on their own will require alternative working methods, but ones that I would suggest are already being used in human creative practice: instead of placing the software in the secondary, and limited role, of assistant, the software can be a viable partner and collaborator, offering independent ideas that extend, provoke, and elevate the individual's creative process.

[3] see https://tinyurl.com/y8r982op

[4] See, for example, a recent BBC news story: https://tinyurl.com/y9hhckm6

References

1. Assayag, G., Rueda, C., Laurson, M., Agon, C., & Delerue, O. (2006). Computer-assisted composition at IRCAM: From PatchWork to OpenMusic. *Computer*, *23*(3).
2. Barlow, C. (2010). Mathematics as the Source of Music Composition. *Proceedings of the 1st International Symposium on Music/Sonic Art*, Baden-Baden, Germany.
3. Bown, O., Carey, B., & Eigenfeldt, A. (2015). Manifesto for a Musebot Ensemble: A platform for live interactive performance between multiple autonomous musical agents. *Proceedings of the International Symposium of Electronic Art*, Vancouver.
4. Card, S., Moran, T., & Newell, A. (1984). *The Psychology of Human-Computer Interaction*, Hillsdale, NJ.
5. Chadabe, J. (1984). Interactive Composing: An Overview. *Computer Music Journal 8*(1), 22–27.
6. Chadabe, J. (1996). The history of electronic music as a reflection of structural paradigms. *Leonardo Music Journal*, 41–44.
7. Cope, D. (1987). An expert system for computer-assisted composition. *Computer Music Journal*, *11*(4), 30–46.
8. Cope, D. (2005). Computer models of musical creativity. Cambridge: MIT Press.
9. Cope, D. (2013). The well-programmed clavier: style in computer music composition. *XRDS: Crossroads, The ACM Magazine for Students, 19*(4), 16–20.
10. Dorfman, L., & Young, D. (1986). Atari ST: Introduction to MIDI Programming. Abacus Software.
11. Ebcioğlu, K. (1988). An expert system for harmonizing four-part chorales. *Computer Music Journal*, *12*(3), 43–51.

12. or Computer Improvisation? A composer's search for intelligent tools in interactive computer music. *Proceedings of the Electronic Music Studies Conference*, 2007.

13. Eigenfeldt, A. (2009). The Evolution of Evolutionary Software: Intelligent Rhythm Generation in Kinetic Engine. *EvoWorkshops* Vol. 9, 498–507.

14. Eigenfeldt, A. (2016). Musebots at One Year: A Review. *Proceedings of the Musical Metacreation Workshop*, Paris.

15. Eigenfeldt, A., Bown, O., Brown, A. R., & Gifford, T. (2016). Flexible Generation of Musical Form: Beyond Mere Generation. *Proceedings of the International Conference on Computational Creativity*, Paris.

16. Eigenfeldt, A (2017). Designing Music with Musebots. *Proceedings of the Fifth Conference on Computation, Communication, Aesthetics and X*, Lisbon, 182–192.

17. Eno, B. (1996) "Generative music". Transcript of talk from Imagination Conference, San Francisco, 8 June 1996. In *Motion Magazine*. Available online: http://www.inmotionmagazine.com/eno1.html.

18. Galanter, P. (2003). What is generative art? Complexity theory as a context for art theory. *Proceedings of the 6th Generative Art Conference*.

19. Galanter, P. (2012). Computational aesthetic evaluation: past and future. *Computers and Creativity.* Springer Berlin Heidelberg, 255–293.

20. Garnett, G. (2001). The aesthetics of interactive computer music. *Computer Music Journal*, *25*(1), 21–33.

21. Hiller, L., & Baker, R. (1964). Computer Cantata: A study in compositional method. *Perspectives of New Music*, 62–90.

22. Koenig, G. (1978). Compositional Processes. *UNESCO Computer Music Workshop*, Aarhus, Denmark. Ottawa: UNESCO.

23. Koenig, G. (1983). Aesthetic integration of computer-composed scores. *Computer Music Journal, 7*(4), 27–32.

24. Kramer, J. (1988). *The time of music: New meanings, new temporalities, new listening strategies.* Schirmer Books.

25. Lewis, G. (1999). Interacting with latter-day musical automata. *Contemporary Music Review*, *18*(3), 99–112.

26. López-Ortega, O. (2013). Computer-assisted creativity: Emulation of cognitive processes on a multi-agent system. *Expert Systems with Applications*, *40*(9), 3459–3470.

27. Pasquier, P., Eigenfeldt, A., Bown, O., & Dubnov, S. (2016). An Introduction to Musical Metacreation. *Computers in Entertainment (CIE) 14*(2).

28. Toole, B. (1996). Ada Byron, Lady Lovelace, an analyst and metaphysician. *IEEE Annals of the History of Computing*, *18*(3), 4–12.

29. Ventura, D. (2015). Mere Generation: Essential Barometer or Dated Concept? *Proceedings of the International Computational Creativity Conference*, Park City.

30. Wiggins, G. (2008). Computer Models of Musical Creativity: A Review of Computer Models of Musical Creativity by David Cope. *Literary and Linguistic Computing 23(*1), 109–116.

31. Xenakis, I. (1971). Formalized Music. Bloomington: Indiana University Press.

32. Winkler, T. (2001). *Composing interactive music: techniques and ideas using Max*. MIT Press.

33. Whitelaw, M. (2004). Metacreation: Art and Artificial Life. MIT Press.

Author Biography

Arne Eigenfeldt is a composer of live electroacoustic music, and a researcher into intelligent generative music systems. His music has been performed around the world, and his collaborations range from Persian Tar masters to contemporary dance companies to musical robots. He has presented his research at conferences and festivals such as the International Computer Music Conference (ICMC), Sound and Music Computing (SMC), the International Conference on Computational Creativity (ICCC), the International Symposium on Electronic Art (ISEA), Creativity and Cognition, EvoMusArt, Generative Art, and New Interfaces for Musical Expression (NIME). He is a professor of music and technology at Simon Fraser University, and is the co-director of the Metacreation Lab.