

Composing for Acoustic Robots - Instant Synthesis For Computer-Controlled Acoustic Instruments Through Live Coding And AI

Alexandros Drymonitis

Cyprus University of Technology

alexdrymonitis@gmail.com

Marinos Koutsomichalis

Cyprus University of Technology

m.koutsomichalis@cut.ac.cy

Submitted: 2024-08-28

Abstract: This paper outlines an ongoing research project on instant composition for computer-controlled acoustic instruments performed by humans. Specifically, it concerns the composition of original works for two computer-controlled instruments, a Yamaha Disklavier and a MIDI-controlled organ, employing LiveLily – a system for live scoring/sequencing through live coding in a Lilypond-like language – and a Recursive Neural Network generating patterns on-the-fly and during performance. Here we review the state-of-the-art underscoring the urgency for research in this subarea, outline a method and present preliminary findings as a proof of concept.

Introduction

This research announcement concerns the 'Composing For Acoustic Robots' (COMACROB) project that commenced in July 2024 and is expected to conclude in a series of original works of experimental music. More specifically, it concerns the use of computer-controlled acoustic instruments, and how these can be used combined with live coding, live scoring, and Artificial Intelligence (AI), together with human performers. The COMACROB is concerned with the three following questions:

Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). Copyright remains with the author(s).

Chroma 2025, 40, 1.

- How can state-of-the-art text-based AI frameworks enhance live coding?
- How can such an AI framework be trained, and how can an effective training dataset be assembled?
- How can live coding and computer-controlled acoustic instruments expand the performance capacity of an instrumentalist?

The ultimate goal is to create a series of short instant compositions for a computer-controlled acoustic instrument, combined with a human performer, that nevertheless does not give the impression of improvisation. Live coding is a celebrated artistic practice that is still under active development. It concerns both the technology used and a manifold of artistic/conceptual approaches. With its own conference series¹ at least one book (Blackwell et al. 2022) and one journal volume (Organised Sound 2023) entirely dedicated to practices thereof, and an abundance of publications in conferences such as New Interfaces for Musical Expression or International Computer Music Conference, it is clear that live coding has already achieved artistic and academic impact. AI music efforts have also seen tremendous development over the past years. Here too we see a dedicated conference² and an abundance of scholarship. Examples of publications include Moutsopoulos et al., who compare adversarial and non-adversarial Long Short-Term Memory (LSTM) Recurrent Neural Network (RNN) music composers, while Ycart and Benetos (2020), Li et al. (2021), Yu et al. (2021) and Min et al. (2022) are similar research projects in the context of the use of LSTM RNN composers. Another example that takes a different approach is *Rave*, where Caillon et al. (2021) created a real-time Variational Autoencoder (VAE) to generate raw audio.

Live scoring is another discipline that has seen significant development and research. Projects such as Inscore (Fober 2012), the bach objects for Max (Agostini 2013), Maxscore (Hajdu 2012), OpenMusic (Bresson 2014), and the Comprovisador (Louzeiro 2018) are a few examples of research in this field. Artistic works that (partly) focus on live scoring include Baird (2005), Eacott (2011) and Drymonitis (2021). One can see that the existing literature on live scoring stands equally high to that of live coding and AI. Moreover, despite live coding having been implemented in various ways and combinations (McClean 2015), from the existing literature one can conclude that live coding has not been extensively applied to controlling acoustic instruments, or live scoring.

¹ International Conference on Live Coding, <https://iclc.toplap.org/>

² AI Music Creativity, <https://aimusiccreativity.org/>

Computer-controlled acoustic instruments is perhaps the least explored field that is included in this research. The Disklavier³, a motorized piano developed by Yamaha, is arguably the most popular of such instruments, with research utilizing it having already taken place (Repp 1997; Willey 2014; Malykhina 2019; Fontana 2024). This is one of the two instruments with which this project is concerned and is privately owned, and located in Athens, Greece. The other is a MIDI-controlled organ that is located in Iceland, at the Frikirkjan church with which the Intelligent Instruments Lab⁴ in Reykjavik, run by Thor Magnusson, collaborates. These two instruments provide a framework that is more familiar to traditional Western-music composers than instruments that are found in a more experimental spectrum, as we will explain further in the section on computer-controlled instruments.

This project also draws on previous research by one of the two authors (Drymonitis 2023a) that focuses on the open-source *LiveLily* system (Drymonitis 2023b), a system for live scoring and live sequencing through live coding in a language inspired by Lilypond (Nienhuys 2003). This previous research concerned the instant synthesis of chorals in the style of Bach, through the use of an RNN built with the *TensorFlow* library (tensorflow 2021), that was trained in the chorals of Bach found in the music21 corpus (Cuthbert 2010). These chorals were exported in the MusicXML format using the provided feature set of music21, and converted to *LiveLily* code through a Python script. Once the AI model was trained, it was invoked through *LiveLily* to generate music through this language. Figure 1 shows a *LiveLily* session. Unlike this previous research, however, in this project, the use of acoustic instruments and the collaboration of contemporary composers are the main focus. Additionally, in this research, AI is tested from a different angle, where symbolic music data is used instead of using a text generator trained on the *LiveLily* language.

It should be noted that *LiveLily* has been chosen instead of the similar *OpusModus*.⁵ This software now has also integrated ChatGPT, trained on the entire documentation of this scoring software, but this use of AI is different than the one intended by this research, where the former will use AI to generate music content, rather than use it as a tutorial feature. Also, *OpusModus* enables real-time feedback while coding a score, but does not aim at actual live scoring. *LiveLily* is accompanied by a software that projects only part of the full score, so an instrumentalist can sight-read the music created, thus it aims at real live scoring. Additionally, it is open-source

³ https://usa.yamaha.com/products/musical_instruments/pianos/disklavier/index.html

⁴ <https://iil.is/>

⁵ <https://www.opusmodus.com/>

and runs on all major Operating Systems, including Linux, whereas *OpusModus* is commercial and does not run on Linux.

Another aspect that is important to this research is transparency, concerning the sources used to train the AI models. At the time of writing, there are several AI content generators for music or other creative fields, where the data used to train these models are unknown to the public, and the entire training process of these models is completely obscure. One of the main goals of this research is to make the names of the composers and their compositions known to the public, be it the audience that will attend the concerts of the instant composition series, or the readers of the literature that will be written for this research, including this paper.

While live-coding, live-scoring, and AI have been individually explored in some depth, to the best of our knowledge there are no other projects combining the three. Accordingly, we believe that it is very timely and urgent to announce our interest to work in this direction and present our findings thus far, and to initiate a creative discourse around related affairs of technical, aesthetic, and contextual nature. Moreover, given that live coders arguably tend to avoid Western-music notation and to approach composition from all sorts of different perspectives (Magnusson 2011b), this project also aims to re-situate the question of Western-music notation within a live-coding context. That is, how to creatively enable state-of-the-art technology in a more traditional compositional process. We hope that this announcement paper will motivate fellow researchers with similar interests to work in these two directions so that findings can be compared at a later stage.

The structure of this paper is as follows: the next section is on the methodology of this research. The subsequent section is on computer-controlled acoustic instruments. A section on the criteria for success of the AI models used in this research then follows, which is an expansion on the methodology. The next section is on technical aspects of the use of AI, with a section on live coding as an interface following. The last section is the conclusions of this paper.

```

1 \score show
2 \score animate showbeat
3
4 \insts Soprano Alto Tenor Bass
5 \Tenor clef bass
6 \Bass clef bass
7
8 \time 4/4
9
10 \bar 1 {
11   \Soprano d''2 f''2
12   \Alto g'2 f'2
13   \Tenor bes2 c'2
14   \Bass g2 a2
15 }
16
17 \bar 2 {
18   \Soprano d''4 d''4 d''4 d''4
19   \Alto f'4 fis'4 g'4 a'4
20   \Tenor d'4 c'4 bes4 a4
21   \Bass bes4 a4 g4 fis4
22 }
23
24 \bar 3 {
25   \Soprano ees''2 d''2
26   \Alto g'4 f'2 g'4
27   \Tenor bes4 c'2 bes4
28   \Bass g4 a4 bes4 g4
29 }
30
31 \bar 4 {
32   \Soprano c''2 c''2
33   \Alto g'2 f'2
34   \Tenor bes2 a2
35   \Bass ees4 c4 f2
36 }
37
1 livelily_bach2.lyv*

```

Fig. 1. A *LiveLily* session with the music of Bach.

Methodology

The methodology applied to the first series of instant composition consists of four stages, outlined in Figure 2. The first stage is the collection of sheet music. As already stated in this paper, the sheet music will be composed by a group of composers who will participate in one of the two projects of this research. At the time of writing, the first series of instant compositions has concluded, with the presentation concert conducted on the 30th of November 2024. Four compositions for two pianos were submitted by the composers Niki Krasaki, Yiannis Sfyris, Klio Barda, and Giorgos Chanos.

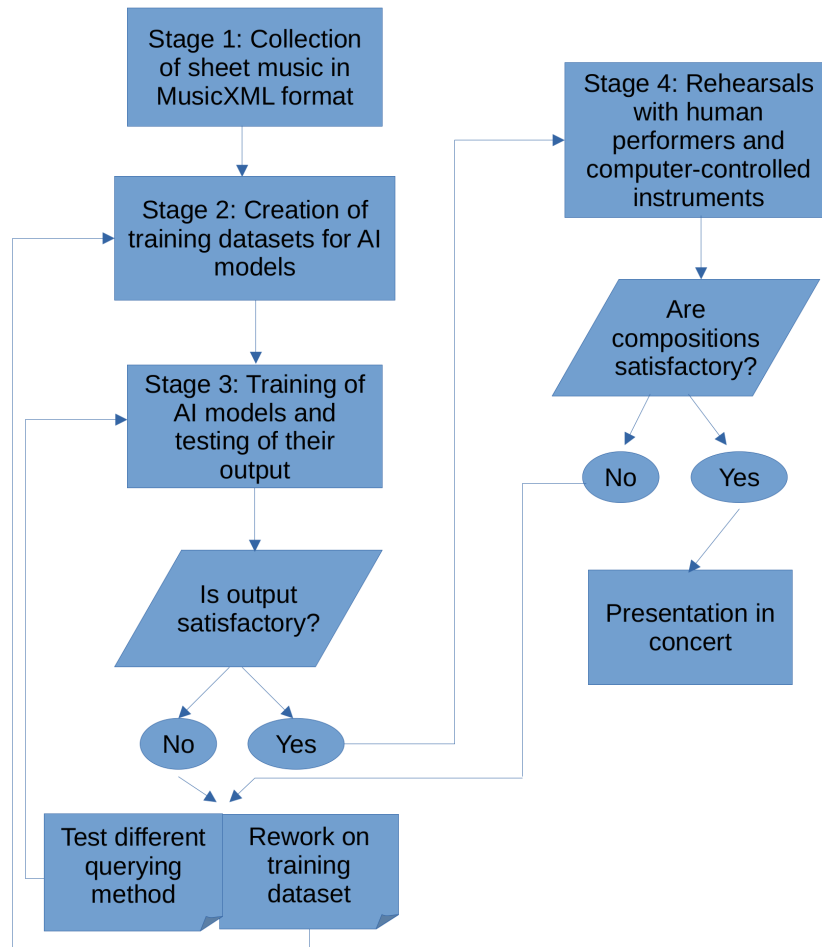


Fig. 2. Flowchart of the research methodology.

Once a large enough corpus of music has been collected, the next stage is to compile one or more training datasets for the AI model that will be used. The composers will provide MusicXML files, which is available by most score engraving systems. When all the music scores have been collected, we will have to decide how to blend the different compositions, to create training datasets. Currently, one training dataset has been assembled, where all four staves are blended, in contrast to the MusicXML format, where each staff is outlined entirely, before the next staff. While this technique has provided satisfactory results, the creation of more datasets needs to be tested further, in case the training corpus is large enough. For the first series of instant compositions, the four compositions provided by the composers mentioned above was not large enough to be separated to smaller datasets. This methodology stage is further analyzed in the "AI at Work" section.

The third stage includes the training of one or more AI models, with the training corpora. As with the training datasets, currently one AI model is trained, which generates music for all staves. The criteria for the success of the AI are mentioned in a forthcoming section. The fourth stage includes rehearsing with the human performers and the computer-controlled instruments.

When all this is achieved, the two series of short compositions will be presented to an audience. As already mentioned, the Disklavier pieces were performed on the 30th of November, 2024, at the Athens Conservatoire⁶, whereas the MIDI-controlled organ pieces were scheduled at the Frikirkjan church in Reykjavik, in mid-April 2025.

Computer-Controlled Acoustic Instruments

This section presents some examples of computer-controlled acoustic instruments.

Bespoke Actuated Instruments

Dan Overholt et al. created a series of actuated acoustic instruments (2011), including a haptic drum, an electromagnetically prepared piano, and a feedback resonance guitar. Steve Mann et al. created a computer-controlled instrument that generates sound from the material world (2008) that is controlled by ice-skates. Angelo et al. present a series of instruments that are actuated through speakers that are attached to them in various ways (2018). Britt et al. created an electromagnetically actuated vibraphone (2012).

Another instrument that fits this section is the Halldorophone (Ulfarsson 2018). This is a cello-like instrument that has both pickups and actuators to create physical feedback on the actual instrument. Even though it is stand-alone, it can be combined with a computer, as it provides jack sockets for its audio input and output, in which case, the computer would intervene in the feedback process.

An Actuated Acoustic Piano

A relevant instrument to the Disklavier, which is used in this research, is the Magnetic Resonator Piano, developed by Andrew McPherson (2010). This is an augmented piano where actuators force the strings to vibrate, enabling infinite sustain, crescendos from silence, harmonics, and other timbres that are not possible with the standard way of playing the piano. Its relevance to this research is the use of a computer-controlled acoustic piano.

⁶ <https://www.athensconservatoire.gr/>

Its difference though is that the Disklavier imitates the standard way of playing the piano by using solenoids to push the hammers on the strings, rather than using other means to force the strings to vibrate.

The Instruments Chosen for this Research

While the above-mentioned instruments are largely bespoke, this project focuses on readily available instruments. The Disklavier is being developed by Yamaha and is commercially available. The MIDI-controlled organ is arguably not a common instrument, nevertheless there are a few such instruments across Europe. The Orgelpark in Amsterdam owns two: the Sauer Organ and the Utopa Baroque Organ (New Baroque Organ 2020). Other computer-controlled organs are located in Reykjavik, at the Frikirkjan and the Hallgrimskirja churches. We intend to use one that is located at the Frikirkjan church.

The reason for choosing instruments that are readily available is that the experimentation of this research is focused on the compositional technique, rather than on instrument-making or on ways to extend/augment instruments. Additionally, aiming to approach the traditional Western-music composers, through a compositional technique that is experimental, choosing instruments like the piano and the organ provides a field that is familiar to the composers, leaving the computer-controlled aspect and the use of their music in an AI context the only parameters not known to them, so they can be the focus.

Criteria for Success of AI

The criterion for the third stage of the methodology of this research is whether the output of the AI model is consistent concerning harmonic content, compared to the training corpus. After a series of experimentation, other musical aspects like rhythm and pitch range were left out of the AI generation process, as, with a training corpus of the size of the one that was made available for the first series of instant compositions, errors on overall duration of notes and extreme pitches in chords that cannot be played were rather common. If this criterion is not met, either the training dataset will have to be re-compiled, or the way an output of the AI model is requested will have to be re-approached.

During this stage for the first series of instant compositions, the format of the training dataset was changed a few times. We have experimented with a few different formats, where data like the beginning of a bar, the meter, the tempo, and articulations were included, but ended up in using a format with notes only, following the Dutch naming system, and two additional words, *sharp* and *flat*. This resulted in a training corpus with 14,784 characters, and a vocabulary of 9 characters. Training the RNN model with this dataset took two hours and twelve minutes, and resulted in a loss of 0.0694.

In addition to the RNN that is used, the possibility of using a Variational Autoencoder (VAE), or a Generative Adversarial Network (GAN) was also considered. Inspired by Tahiroglu (2021), the latent space can help in grouping similar outputs, following the criterion outlined in this paragraph, and querying outputs by using these latent space mappings. We attempted to train a VAE on the corpus we assembled, but the results were almost completely repetitive, where the network would output the same character constantly. We believe that the reason for this is the size of the training data. For the second series of instant compositions, if we manage to collect enough data, these AI architectures will be re-visited.

The rehearsals, which comprise the fourth stage of our research methodology, were initially separate, where the human performers played their part without the computer intervening physically. This is due to the nature of the *LiveLily* system, whose interactivity is mostly unknown to classically trained musicians. Rehearsing separately without the computer-controlled part enabled the performers to familiarize themselves with this system. After a few rehearsals with the performers only, we moved to rehearse with the actual Disklavier, with the computer in control. In these rehearsals we could hear the complete compositions provided by the composers, and the AI generated versions of them. This stage helped determine whether the AI model performed at a satisfactory level. Where this was not the case, adjustments were made, either to the training dataset, in which case the AI models will have to be retrained, or to the structure of the model and/or the way the model's predictions are queried and used. Hence, the third and fourth stages of the methodology of this research might force a return to either from the second or third stage. For the first series, we introduced a percentage within which each note is likely to change, based on the output of the AI model. This way, we were able to control how much of the original composition we would hear, and how much the AI model would interfere.

The AI at Work

In Drymonitis (2023a), all AI-related work was done in Python with *TensorFlow*. This project still uses *TensorFlow* for generating *LiveLily* code, as was the case of Drymonitis (2023a), but the approach is different in the sense that the RNN used is not trained on *LiveLily* code as a text generator, but on symbolic music data from the compositions of the four composers. After trying out different RNNs for text, it appeared that, in contrast to Drymonitis (2023a), the training corpus of the current project is not sufficient to train a text generator to a high enough standard. After researching different types of RNNs that could work, we found that the system developed

for the Classical-Piano-Composer⁷ provides satisfactory results, even though at the time of writing this project has not been updated in five years.

The Classical-Piano-Composer aims at creating tonal music, trained on 92 songs that are played on a single piano. It parses MIDI files with the music21 module, and assembles a training dataset by extracting note and chord information. For our research, we initially expanded on the elements extracted, by adding rests, durations, tuplets, and dots in durations. Our training corpus, being much smaller than the one used in the Classical-Piano-Composer, provided more information than the RNN could handle in a meaningful manner. This led us to discard this extra information and focus on the notes data only. For the needs of our research, we split each composition we used to four parts, one for each hand of each performer, the computer-controlled instrument and the human performer. The RNN is trained on a dataset with alternating staves, where for one bar, all four staves are added to the dataset sequentially, and this is repeated for all bars of all four compositions. This process is similar to flattening 2-dimensional data, like image pixels. The output of the RNNs is translated to *LiveLily* code, and "typed" by the Python script in *LiveLily*, utilizing the remote typing feature of this software.

Currently, the AI part of this project is done in Python, but future plans include integrating this to *LiveLily*, through *ofxTensorFlow2*.⁸ This is an openFrameworks add-on for loading and running Machine Learning (ML) models trained with the *TensorFlow 2 ML* library. The reason for this choice is that *LiveLily* is written in openFrameworks, and utilizing this add-on enables the integration of the code-generating component of the AI into *LiveLily* itself. Integrating the AI into *LiveLily* allows for less configuration steps, and it further enables all the technology used to be run on the same software, thus providing a system that is less prone to error. The building and training of the AI model still needs to be done in Python, but this is a separate process where the software (*Python* and *LiveLily*) does not need to run concurrently. Once the AI model is trained and saved, it will be loaded in *LiveLily* and run there.

Live Coding as Interface

Live coding is known to be slow at evolving throughout a performance (Magnusson 2011). Nevertheless, following McLean's paradigm on economical syntax (2014), *LiveLily* provides syntactic sugar on top of the Lilypond language (Drymonitis 2023b), enabling faster typing of rhythmic and melodic patterns.

⁷ <https://github.com/Skuldur/Classical-Piano-Composer/>

⁸ <https://github.com/zkmarlsruhe/ofxTensorFlow2>

Syntactic Sugar

The syntactic sugar *LiveLily* provides includes repetitions of notes, chords, or bars by using the multiplication feature. The line below demonstrates the repetition of notes and chords, which results in the score of Figure 3.

```
c''16*4 <g' a'>4*2 ees'8*2
```

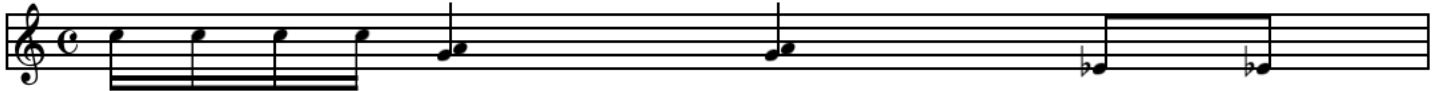


Fig. 3. Score produced using LiveLily's syntactic sugar.

Another use of the multiplication feature concerns the creation of loops. In *LiveLily*, it is possible to create various bars separately, and then create a loop with any of these. If, for example, one creates bars with incrementing numbers (e.g. 1, 2, 3, etc.), one can create a loop by writing the following line.

```
\loop loopname {\1 \2 \3}
```

If one wishes any of these bars to be repeated more than once within a loop, one can use the multiplication feature to specify the number of repetitions of this bar within the loop. The following line creates a loop where bar 1 is repeated twice, and the bars 2 and 3 are played once.

```
\loop loopname {\1*2 \2 \3}
```

Another feature of *LiveLily* is the extraction of bar data from an existing bar. If, for example, one defines two instruments, and defines one bar for both, then one can define a second bar where new data can be written for one of the instruments, while the other instrument can repeat what has been defined in the first bar. The following code chunk demonstrates this.

```
\bar 2 {  
  \inst1 c''16*4 <g' a'>4*2 ees'8*2  
  \inst2 \1  
}
```

In the code example above, the user defines a bar named 2 with a pattern for `\inst1`, while `\inst2` copies its pattern from a previously defined bar, named `\1`. All this, combined with the automatic insertion of certain characters by the *LiveLily* editor, like closing angle and curly brackets, makes typing in this editor and language rather fast.

The Contribution of AI

While economical syntax can help in a live coding session, the AI will also make performing with live coding easier. Invoking the AI model during the performance reduces a great deal of typing from the coder, as this is handled by the AI mechanism. When the model has provided a prediction, this is typed in *LiveLily* a lot faster than what an average live coder types. By frequently invoking the AI model, it is possible and desirable to let the AI do all the content creation, and the live coder will determine which bars will be played, and in which sequence, or even let the AI content be played in the sequence it is created.

All the live coder needs to do is to provide the name of the composer whose music the AI model will use as a prompt. To refine the control over the AI-generated content, we introduced a *difficulty* coefficient, plus the ability to provide one composer for the computer-controlled instrument, and another for the human performer. This coefficient concerns a mapping on each bar of all compositions, based on the density of the notes. By using this, we have control over the flow of the instant composition, and we can aim at each performer separately, as they do not all share the same sight-reading capabilities.

Figure 4 illustrates a session of this research. The top pane is where the live coder types, and the bottom pane is where the AI model types. Using live coding enables us to communicate our intentions. Before the concert, we provide a few explanations as to the meaning of what the live coder types, so the audience can follow his/her train of thought, even if they are not familiar with computer code. In Figure 4, the bottom line in the top pane, sets Niki Krasaki as the composer for the Disklavier and Yiannis Sfyris for the performer. This is done with their initials, here "ks". The numbers 3 and 1 determine the difficulty for each part, in a scale from 0 to 6. The line above that, sets the percentage for the AI-generated notes to pass to 40.

The screenshot shows the LiveLily application window. On the left is a code editor with the following code:

```

5 \score { animate
6 \score { tempo show
7
8 \Primo1 sendto
9 \Primo2 sendto
10 \Secondo1 sendto
11 \Secondo2 sendto
12 \Primo1 fullscreen on
13 \Primo1 accoffset 2.0
14 \group \Primo1 \Primo2
15 \group \Secondo1 \Secondo2
16
17 \listmidiparts
18 \openmidiport 2
19 \Primo1 midichan 1
20 \Primo2 midichan 1
21 \Secondo1 midichan 2
22 \Secondo2 midichan 2
23
24 \use pd
25 \pd setup 9010
26 \function sendbeat {
27   \pd send /beat \beatcount
28 }
29
30 \sendbeat bind beat
31 \py send /percent 40
32 \py send /make "ls" 3
33
34
35
36
37
38
39
40
41 \loop nn5-8 { \nn5 \nn6 \nn7 \nn8 } \nn5-8
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

On the right is a musical score with three staves: Violino 1, Violino 2, and Cello. The score starts with a tempo of 70 and a 4/4 time signature. The first measure shows a treble clef for Violino 1 and Violino 2, and a bass clef for Cello. The notes are: Violino 1 (G4, A4, B4, C5), Violino 2 (G4, A4, B4, C5), and Cello (G2, A2, B2, C3).

Fig. 4. Score produced by the AI model.

Other LiveLily Features

Apart from the features mentioned in this section, *LiveLily* includes some other features that can make live coding fast. These are muting, unmuting, and soloing instruments, with the `\mute`, `\unmute`, and `\solo` commands. By using these features during a performance, it is possible to create variations on the music in a very responsive way.

Conclusions

This paper has presented the concept, software, and methodology of the Postdoctoral research titled "Composing for Acoustic Robots". While this research utilizes well-established practices within the greater field of music, like live coding, live scoring, and AI, it combines them in a novel way, while at the same time approaching traditional Western-music composition. Its originality also lies in the use of computer-controlled acoustic instruments through the use of live coding.

While being a research in progress, this paper aims at initiating a discourse around the research fields that comprise this project, namely, live coding, live scoring, AI, and computer-controlled instruments combined. We believe that it is both timely and urgent to announce our research orientation, as well as report our findings thus

far, so as to motivate fellow researchers that find themselves in a similar spectrum to continue working in this direction. We hope that this leads to more research and literature on this field becoming available in future.

The instruments used for this research are a Disklavier piano and a MIDI-controlled organ. These instruments were chosen so as to provide a framework that is familiar to the participating composers, since the focus of this research is on the composition techniques, and not on extending/augmenting instruments. With traditional Western-music composition and notation being arguably avoided by live coders, we aim to re-approach the question of Western-music score creation through live coding, enabling the use of state-of-the-art technology in a more traditional compositional process.

Acknowledgements

This research is funded by the Cyprus University of Technology, as part of its Post-Doctoral Research Programme.

References

- Agostini, A., & Ghisi, D. (2013). Real-Time Computer-Aided Composition with bach. *Contemporary Music Review*, 32. <https://doi.org/10.1080/07494467.2013.774221>
- Ângelo, T., Penha, R., Gomes, J., & Rebelo, P. (2018, June). Actuated Musical Instruments - A State of the Art. *International Conference on Live Interfaces (ICLC)*.
- Baird, K. C. (2005). Real-Time Generation of Music Notation via Audience Interaction Using Python and (GNU) Lilypond. *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, 240–241. <https://doi.org/10.5281/zenodo.1176695>
- Blackwell, A. F., Cocker, E., Cox, G., McLean, A., & Magnusson, T. (Eds.). (2022). *Live Coding: A User's Manual*. The MIT Press.
- Bresson, J. (2014). Reactive Visual Programs for Computer-Aided Music Composition. *Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC*, 141–144. <https://doi.org/10.1109/VLHCC.2014.6883037>
- Britt, N. C., Snyder, J., & McPherson, A. (2012). The EMvibe: An Electromagnetically Actuated Vibraphone. *Proceedings of the International Conference on New Interfaces for Musical Expression*. <https://doi.org/10.5281/zenodo.1178221>
- Caillon, A., & Esling, P. (2021). *RAVE: A Variational Autoencoder for Fast and High-Quality Neural Audio Synthesis*. <https://arxiv.org/abs/2111.05011>
- Cuthbert, M. S., & Ariza, C. (2010). Music21: A Toolkit for Computer-Aided Musicology and Symbolic Music Data. *Proceedings of the 11th International Society for Music Information Retrieval Conference*, 637–642. <https://doi.org/10.5281/zenodo.1416114>
- Drymonitis, A. (2023a). Creating Music Scores Live Through Interacting with a Character-level Text Generator. *Live Interfaces Journal*, 1. <https://doi.org/10.60543/liveinterfacesjournal.v1i1.9144>
- Drymonitis, A. (2023b). LiveLily: An Expressive Live Sequencing and Live Scoring System Through Live Coding With the Lilypond Language. *Proceedings of the International Conference on New Interfaces for Musical Expression*. <https://doi.org/10.5281/zenodo.11189176>

Drymonitis, A., & Chatzopoulou, N. (2021). Data Mining / Live Scoring – A Live Performance of a Computer-Aided Composition Based on Twitter. *Proceedings of the 2nd Joint Conference on AI Music Creativity*. <https://doi.org/10.5281/zenodo.5137948>

Eacott, J. (2011). Flood Tide See Further: Sonification as Musical Performance. *Proceedings of the International Computer Music Conference, (ICMC)*, 69–74.

Fidon, H. (Ed.). (2020). *Orgelpark Research Report 5/1*. VU University Press. <https://www.orgelpark.nl/zip/extracted/web-5/1/nl/>

Fober, D., Orlarey, Y., & Letz, S. (2012, May). INScore An Environment for the Design of Live Music Scores. *Proceedings of the 2021 Linux Audio Conference, LAC*.

Fontana, F., Järveläinen, H., Papetti, S., & Pra, Y. (2024). Acoustic Cues of Keyboard Mechanics Enable Auditory Localization of Upright Piano Tones. *The Journal of the Acoustical Society of America*, 156, 164–175. <https://doi.org/10.1121/10.0026484>

Hajdu, G., & Didkovsky, N. (2012). Maxscore - Current state of the art. *Proceedings of the International Computer Music Conference, (ICMC)*, 156–162.

Landy, L., Xambo Sedo, A., Gerard, R., & Magnusson, T. (Eds.). (2023). *Organised Sound: Live Coding Sonic Creatives*. Cambridge University Press.

Li, G., Ding, S., Li, Y., & Zhang, K. (2021). Music Generation and Human Voice Conversion Based on LSTM. *MATEC Web Conf.*, 336, 06015. <https://doi.org/10.1051/mateconf/202133606015>

Louzeiro, P. (2018). The Comprovisador's Real-Time Notation Interface (Extended Version). In M. Aramaki, M. E. P. Davies, R. Kronland-Martinet, & S. Ystad (Eds.), *Music Technology with Swing* (pp. 489–508). Springer International Publishing.

Magnusson, T. (2011a). Algorithms as Scores: Coding Live Music. *Leonardo Music Journal*, 21, 19–23. https://doi.org/10.1162/LMJ_a_00056

Magnusson, T. (2011b). The IXI Lang: A SuperCollider Parasite for Live Coding. *International Computer Music Conference (ICMC)*, 503–506.

Malykhina, I. (2019). Integration of Hybrid Musical Instruments into Russian Piano School. *Integration of Education*, 23, 137–149. <https://doi.org/10.15507/1991-9468.094.023.201901.137-149>

Mann, S., Janzen, R., & Lo, R. (2008). Hyperacoustic Instruments: Computer-Controlled Instruments That Are Not Electrophones. *2008 IEEE International Conference on Multimedia and Expo*, 89–92. <https://doi.org/10.1109/ICME.2008.4607378>

McLean, A. (2014). Making programming languages to dance to: live coding with tidal. *Proceedings of the 2nd ACM SIGPLAN International Workshop on Functional Art, Music, Modeling & Design*, 63–70. <https://doi.org/10.1145/2633638.2633647>

McLean, A. (2015). Reflections on Live Coding Collaboration. *Proceedings of 3rd Conference on Computation, Communication, Aesthetics and X (XCoAx)*, 213–220.

McPherson, A. (2010). The Magnetic Resonator Piano: Electronic Augmentation of an Acoustic Grand Piano. *Journal of New Music Research*, 39(3), 189–202. <https://doi.org/10.1080/09298211003695587>

Min, J., Liu, Z., Wang, L., Li, D., Zhang, M., & Huang, Y. (2022). Music Generation System for Adversarial Training Based on Deep Learning. *Processes*, 10(12). <https://doi.org/10.3390/pr10122515>

Mots'oezli, M., Bosman, A. S., & De Villiers, J. P. (2023). Comparison of Adversarial and Non-Adversarial LSTM Music Generative Models. In K. Arai (Ed.), *Intelligent Computing* (pp. 428–458). Springer Nature Switzerland.

Nienhuys, H.-W., & Nieuwenhuizen, J. (2003). Lilypond, a System for Automated Music Engraving. *Proceedings of the XIV Colloquium on Musical Informatics (XIV CIM 2003)*.

Overholt, D., Berdahl, E., & Hamilton, R. (2011). Advancements in Actuated Musical Instruments. *Organised Sound*, 16, 154–165. <https://doi.org/10.1017/S1355771811000100>

Repp, B. (1997). Acoustics, Perception, and Production of Legato Articulation on a Computer-Controlled Grand Piano. *The Journal of the Acoustical Society of America*, 102, 1878–1890. <https://doi.org/10.1121/1.420110>

Tahiroğlu, K., Kastemaa, M., & Koli, O. (2021). GANSpaceSynth: A Hybrid Generative Adversarial Network Architecture for Organising the Latent Space using a Dimensionality Reduction for Real-Time Audio Synthesis. *Proceedings of the 2nd Joint Conference on AI Music Creativity*, 10. <https://doi.org/10.5281/zenodo.5137902>

TensorFlow Developers. (2021). *TensorFlow* (v2.5.0) [Computer software]. Zenodo. <https://doi.org/10.5281/zenodo.4758419>

Úlfarsson, H. (2018). The Halldorophone: The Ongoing Innovation of a Cello-Like Drone Instrument. *Proceedings of the International Conference on New Interfaces for Musical Expression*, 269–274. <https://doi.org/10.5281/zenodo.1302579>

Willey, R. (2014). The Editing and Arrangement of Conlon Nancarrow's Studies for Disklavier and Synthesizers. *Music Theory Online*, 20. <https://doi.org/10.30535/mto.20.1.8>

Ycart, A., & Benetos, E. (2020). Learning and Evaluation Methodologies for Polyphonic Music Sequence Prediction With LSTMs. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28, 1328–1341. <https://doi.org/10.1109/TASLP.2020.2987130>

Yu, Y., Srivastava, A., & Canales, S. (2021). Conditional LSTM-GAN for Melody Generation from Lyrics. *ACM Trans. Multimedia Comput. Commun. Appl.*, 17(1). <https://doi.org/10.1145/3424116>