Interactive Digital Multimedia Techniques 2025
Technical report for the project
Sequential Drum, Reimagined

Yorgos Velissaridis

Abstract

This report details the design, implementation, and evaluation of "The Sequential Drum, Reimagined," a digital musical instrument (DMI) designed to address the disparity between musical understanding and technical dexterity in Western Art Music (WAM). Building upon Max Mathews' Sequential Drum, this project utilises a Python-based script to offload pitch execution to a computer system while granting the performer precise control over timing, dynamics, and articulation via a remapped MIDI interface. Timbre is also liberated, since the system confers full control over what synthesised or VST sound will be used for each note, phrase and register of a music piece. Through the performance of works by Bartók and Bach, the project demonstrates that decoupling pitch selection from performance expression can significantly accelerate repertoire acquisition and allow for sophisticated musical interpretation without the prerequisite of virtuoso motor skills.

1 Introduction

In the domain of Western Art Music (WAM) performance, the execution of a pre-determined note sequence is the foundational requirement. However, for many musicians, the technical demands of executing these pitches accurately can become a barrier to expressive performance. A disparity often exists where a musician can audiate the desired phrasing, timing, and articulation internally, yet lacks the specific motor dexterity to realise that sound in real-time.

This project aims to bridge that gap. The central research question guiding this work is: *What are the affordances of a digital instrument that offloads the requirement of having to play the right notes of a music score?* By shifting the performer's cognitive load from pitch accuracy to temporal and dynamic expression, this project proposes a new paradigm for instrumental practice and performance—one that prioritises "conducting" the sound over mechanical reproduction.

2 Background and Related Work

The core concept of this project is a reimagining of the "Sequential Drum," a system proposed by Max Mathews in the early era of computer music [1]. Mathews envisioned a system where a performer strikes a drum to advance through a score stored in computer memory. In his view, the computer could handle the precise data of pitch, while the human provided the nuance of timing and amplitude [3]. Additionally, Mathews built a custom surface for playing this instrument, where the X and Y axes controlled timbre-related parameters.

While Mathews' work laid the groundwork, modern Digital Audio Workstations (DAWs) and scripting environments offer new possibilities for this interaction model. The field of New Interfaces for Musical Expression (NIME) has long debated the relationship between control and intimacy in instrument design. Wessel and Wright argue that "intimate" control requires low latency and a high degree of

mapping between gesture and sound [5]. This project seeks to maintain that intimacy by ensuring that every note event is directly triggered by a human action, avoiding the detachment often felt in automated playback systems.

Furthermore, Rodger and colleagues posit that a "good" musical instrument is defined by its specificities and the ecology of processes it enables [4]. By reviving the Sequential Drum with modern tools, this project investigates the specific ecology of "performance without pitch-worry," analysing how it alters the musician's relationship to the score.

3 System Design and Implementation

The design process focused on translating static music notation into a fluid, performable timeline. The technical architecture is built upon a Python script serving as a middleware between a standard keyboard MIDI controller and a sound generator (DAW/VST).

3.1 The Parser and Logic Architecture

The core of the system is a Python script utilising the `music21` library to parse MusicXML files. The choice of MusicXML allows for the importation of complex, polyphonic scores from standard notation software.

The parsing logic separates the score into two distinct streams: Left Hand (LH) and Right Hand (RH). These streams are flattened into linear timelines stored in memory. A pointer index tracks the current position in each stream. When a trigger is received, the system retrieves the pitch data at the current index, sends the MIDI message, and advances the pointer.

A significant technical challenge involved handling complex musical notations to ensure musical fluidity:

- **Grace Notes:** In standard sequential processing, grace notes can disrupt the rhythmic flow if treated as standard beats. The system logic identifies grace notes and treats them as "wildcard" events. This allows them to be triggered rapidly by any key in the trigger zone, facilitating gestural sweeps without requiring strict 1-to-1 key mapping.
- **Chordal Logic:** To handle polyphony, the system differentiates between single notes and vertical structures. The LH and RH streams trigger their respective note lists independently. This separation allows for intricate phrasing—such as staggering the entry of a melody against an accompaniment—that would be impossible with a global "next event" trigger.
- **Tie Handling:** To prevent the unnatural re-triggering of tied notes, the parser filters out "stop" ties. This ensures that a note is only triggered at its onset, preserving the legato line essential for WAM performance.

3.2 The Physical Interface and Mapping

The instrument utilises a standard MIDI keyboard (a Korg Monologue in the prototype) remapped to serve as a timeline controller rather than a pitch selector. This is presented as an alternative to Mathew's original design of an X-Y electronic surface. We believe a keyboard is a better interface because it utilises finger dexterity (rather than relying on batons, which Mathews also experimented with) and allows for the transfer of fingering knowledge from other keyboard and string instruments. It is notable that the X-Y surface of Mathews, which allowed for timbral manipulations can be replicated with a keyboard interface, using something like McPherson's Touch Keys [2].

1. **Trigger Zones:** The keyboard is divided into zones. Specific keys (e.g., D through A) are assigned to advance the RH and LH pointers. Using multiple keys for a single function allows the performer to use multiple fingers to execute rapid passages (trills or fast runs, that would be fatiguing on a single button), as well as legato playing, and other articulation nuances that require multiple fingers. Five keys are assigned to each hand, to allow easy triggering with the five fingers of each hand.
2. **Navigation and Error Correction:** To transform the system from a novelty into a viable practice tool, navigation commands were essential. Dedicated keys were mapped to "Bar Forward," "Bar Back," and "Restart Bar."

3. **Auto-Sync:** A recurring issue in split-hand sequential instruments is the de-synchronisation of the two pointers. An "Auto-Sync" feature was implemented that realigns the hand pointers at bar lines. If one hand crosses a bar line, it "pulls" the other hand to the start of that measure, preventing the performer from becoming effectively lost in the sequence.

3.3 Timbre and Sound Design

While the architecture is timbre-agnostic, the prototype migrated from utilising `fluidsynth` to integrating with Ableton Live via virtual MIDI ports. This opened the door for synth-based electronic interpretation.

A critical technical adjustment was required for DAW integration. Early iterations used a "Round Robin" MIDI channel rotation to handle polyphony (preventing the cutting off of release tails). However, many VSTs in Ableton do not support multi-channel input for a single patch, leading to voice-stealing issues. A "Single Channel Mode" was implemented to force all output to Channel 1, ensuring stability with complex VSTs like Kontakt or Karplus-Strong synthesisers, at the cost of some polyphonic overlap.

4 Evaluation: The Concert Performance

The system was evaluated through a final concert performance featuring three distinct pieces, chosen to test different capabilities of the instrument. The instrument was played by me, Yorgos Velissaridis, while timbral manipulations were handled by Pablo Tablas.

4.1 Repertoire Selection

- **Bartók, Mikrokosmos No. 32:** A simple two-voice piece in Dorian mode. This served as a proof-of-concept for the split-hand logic, demonstrating the independence of the two trigger streams.
- **J.S. Bach, Allemande from Partita No. 2:** Originally for solo violin, this monophonic piece was selected to highlight timbral capabilities. Freed from the difficulty of intonation and bowing, the performance focused on drastic timbre variations using a Karplus-Strong synthesiser, demonstrating interpretive adjustments via parameter modulation during playback.
- **Bartók, Mikrokosmos No. 148:** The finale was a complex Bulgarian dance featuring irregular rhythms and varying textures, from dense to sparse ones. This served as the "stress test" for the system.

4.2 Reflections on Performance

The project achieved a significant level of success, with parts of the performance working convincingly. The most striking metric of success was the reduction in practice time. *Mikrokosmos No. 148* is a technically demanding piece that might typically require months of practice to master mechanically. With this instrument, I was able to reach a performance-ready standard after approximately one week of practice.

The experience confirmed the hypothesis: offloading pitch processing frees up mental space. The performance felt less like "playing" in the traditional sense and more like "conducting" or real-time editing. The cognitive load shifted largely to rhythm, dynamics, and the physical gesture of triggering. However, this did not remove from my internal awareness of the notes, as I was able to hear the notes of the piece before playing them quite well during the performance, which occurred for the other pieces as well.

5 Critical Analysis and Discussion

5.1 Theory vs. Practice

Theoretically, the Sequential Drum promises total freedom. In practice, however, the instrument introduces its own constraints, aligning with the view of [4] that specificities define an instrument. The primary constraint discovered was "Error Propagation."

In a traditional piano performance, hitting a wrong note is a momentary error; the piece continues. In a sequential instrument, missing a trigger or double-triggering shifts the pointer, causing the *next* note to be the wrong one. This creates a cascading error effect where the performer is playing the future notes in the present time.

To mitigate this, the "patchwork solution" of re-synchronising at bar lines was implemented. While effective, this highlights a friction between the theoretical ideal of the instrument (seamless flow) and the practical reality of digital logic. It transforms the instrument from a pure playback device into a "navigational" device, where the performer must be aware of their location in the data structure as much as the musical structure.

In the future this can be better handled by allowing for some degree of error on the performers side by implementing rule or data-based methods to compensate for them.

6 Skills Gained

This project facilitated the acquisition of distinct technical and creative skills:

- **Python for Real-Time Media:** Gained proficiency with the `mido` library for MIDI event handling and `music21` for data parsing, moving beyond simple scripting to managing real-time event loops.
- **Algorithmic Logic:** Developed skills in handling asynchronous data streams (LH vs RH pointers) and error handling (try/except blocks to prevent performance crashes).
- **System Integration:** Learned to bridge the gap between code and commercial DAWs using virtual ports and understanding VST voice allocation limitations.
- **Instrument Design:** Gained insight into the iterative process of mapping. The realisation that "standard" mappings (1 key = 1 note) are insufficient for complex textures led to the implementation of "wildcard" triggers and zone-based inputs.

7 Future Directions

The current iteration of the instrument focuses on the temporal dimension of performance. The next logical step is to integrate the timbral dimension more deeply. Future work will involve mapping unused controller knobs to VST parameters, allowing the performer to morph sound design dynamically.

Furthermore, I propose a concept of "Pre-Orchestration." Specific phrases or registers in the XML score could be tagged to trigger specific MIDI channels or timbre changes automatically. This would allow the performer to realise the common pedagogical metaphor (e.g., "play this phrase like a flute," "this section is the brass") as a literal sonic reality, further blending the roles of performer, conductor, and sound designer.

References

[1] Max V Mathews and Curtis Abbott. The sequential drum. *Computer Music Journal*, 4(4):45–59, 1980.

[2] Andrew McPherson. Touchkeys: Capacitive multi-touch sensing on a physical keyboard. In *Proceedings of the international conference on New Interfaces for Musical Expression*, 2012.

[3] Curtis Roads and Max Mathews. Interview with max mathews. *Computer Music Journal*, 4(4): 15–22, 1980.

[4] Matthew Rodger, Paul Stapleton, Maarten Van Walstijn, Miguel Ortiz, and Laurel Pardue. What makes a good musical instrument? a matter of processes, ecologies and specificities. In *Proceedings of the international conference on New Interfaces for Musical Expression, NIME 2020*, pages 484–490, 2020.

[5] David Wessel and Matthew Wright. Problems and prospects for intimate musical control of computers. *Computer music journal*, 26(3):11–22, 2002.