Terry Riley's "In C" for Mobile Ensemble

David B. Wetzel
Loyola University Chicago
Chicago, IL
dwetzel@luc.edu

Griffin Moe
Independent Scholar
Chicago, IL
me@griffinmoe.com

George K. Thiruvathukal
Loyola University Chicago
Chicago, IL
gthiruvathukal@luc.edu

ABSTRACT

This workshop presents a mobile-friendly Web Audio application for a "technology ensemble play-along" of Terry Riley's 1964 composition *In C*. Attendees will join in a reading of *In C* using available web-enabled devices as musical instruments. We hope to demonstrate an accessible music-technology experience that relies on face-to-face interaction within a shared space. In this all-electronic implementation, no special musical or technical expertise is required.

1 Introduction

In C is meant to be accessible. Terry Riley's 1964 masterwork consists of fifty-three short *melodic patterns* and a set of rules for performance. [5] Instrumentation is left open, as is the number of players. *In C* requires careful listening and close interaction between participants in a *shared space*. A performance, lasting roughly 45 to 90 minutes, is a product of the collective mood of the participants, and no two readings are alike. As Robert Carl put it, "*In C* is a piece of software ... a series of rules and predefined relationships that execute a task; the user can then customize input and tweak aspects of the rules and relations to produce a product that is regarded as personal" [2].

Our implementation extends the accessibility of *In C* to those without instrumental training or with limited musical skill. Rather than worrying about reading and playing notes correctly, participants focus on elements stressed most by the composer: listening, responding, and making decisions about when or whether to play, when to move forward, or which options to apply. Full participation is therefore open (so far) to anyone that can see, hear, and operate the click- or touch-driven user interface.

The initial prototype, written in Max [1], began as a group project for an introductory digital music course. Under the banner of LUTE (Loyola University Technology Ensemble), it powered numerous open readings for participants of all ages and backgrounds, including play-alongs at Chicago's Thirsty Ears classical street festival. However, the Max implementation presented prohibitive technical hurdles (e.g., downloading and installing on participants' computers) that demanded a more accessible (mobile) technology solution.

2 Using Tone.js

Our earlier prototype depended on the ability within Max to align musical events to a precise metrical grid in an interactive context. Max is built for this, but the Web, generally, is not. However, the Web Audio framework Tone.js offers a "global transport for synchronizing and scheduling events" that is not only accurate but allows for precise scheduling of events along a timeline using musical syntax [3].

Our web app uses Tone.js for all sequencing and synthesis tasks. A melodic pattern is stored as an array of note objects defining time, pitch, and duration, which is then instantiated as a Tone.js "Part". Each Part is only scheduled on the Transport timeline in response to a user "play" action. Therefore, participants can play the individual patterns at any time, or repeatedly, in sync with the Transport.

3 Web App for In C

Our web app (Figure 1) is written in standard HTML/CSS/JS to be broadly compatible with modern browsers, especially on mobile devices.

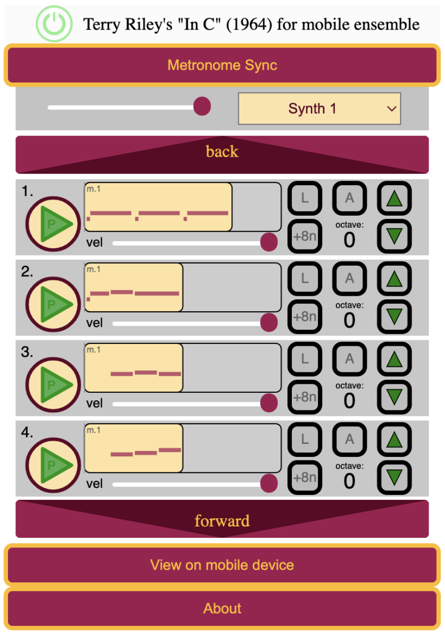

Figure 1: The *In C* Application: lute.luc.edu/InC

While the Tone.js framework is invoked to handle sound generation and musical sequencing tasks, the graphical user interface is largely designed in `P5.js` [4].

3.1 Context Menus and Controls

The app has several context menus that give easy access to important functions or information. At the top of the screen is a familiar "power button" icon that starts or stops the Tone Transport.

Metronome Sync: This section contains controls for a metronome click, 'C' ostinato, and metronome synchronization (Figure 2). The tempo slider only temporarily speeds up or slows down the Transport to allow synchronization with others.

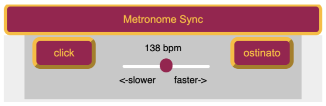

Figure 2: The *In C* Application: Metronome Sync

Main Volume and Synth Selector: Above the sequence player, a slider controls the main output volume. A drop-down menu gives access to a limited set of synthesizer timbres.

View on Mobile Device: Contains a QR code to share or for transferring to a mobile device from a laptop.

About: Contains background information, instructions, and credits for the contributors to this project.

3.2 Sequence Players

All 53 melodic patterns are pre-loaded, but only four are visible at a given time. The forward button progresses to the next pattern. Patterns are rendered in a familiar "piano roll" style (with animated play-head, Figure 3), allowing all patterns to scale to a standard size regardless of length. The sequence player section has the following individual controls for each pattern:

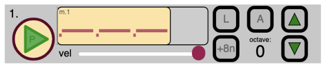

Figure 3: The *In C* Application: Sequence Player

P (Play): Play the pattern on the next available beat.

L (Loop): Loop the pattern indefinitely when played.

A (Rhythmic Augmentation): Play the pattern half speed.

+8n (8th Note Shift): Pattern starts one 8th-note later.

Octave Transposition. Shift or down +/- 3 octaves.

Vel (velocity). adjust the attack volume for notes in the pattern (does not affect continuous volume)

4 Instructions for Play-Along

Setup: Once the app is loaded, its functionality can be quickly verified by starting the Transport (power button) and playing a sequence. Before a performance, all participants must synchronize their metronomes. For this task, one device is designated as "conductor" and its metronome click is turned on (as loud as possible). Synchronization to the conductor requires careful listening and the built-in click. With the click running, use the tempo slider to either speed up or slow down the tempo (+/- 10%) until it comes into phase with the conductor (release the slider to return to the default tempo). This is done individually for all participants and may take some time. We have found that the exercise of sync'ing by ear is more valuable than the convenience that might come from an automated approach.

Once synchronization is complete, all patterns will lock to the same rhythmic grid. Participants should take a moment to explore the various patterns and sounds before performance begins.

Performance: Following setup, performance proceeds as outlined in the score. All participants progress through the full set of 53 patterns, in order, but not necessarily in unison. It is helpful to have a designated conductor to signal transitions and keep the group together (within 2 or 3 patterns of one another). Each participant is free to play, not play, repeat, loop, shift octaves, change timbres, etc., as they see fit, listening carefully to those around them and to the musical texture that develops as a whole.

A "mobile ensemble play-along" of *In C*, like any other performance of this work, will vary greatly in overall duration. A typical performance in this implementation will take roughly 45 minutes. The performance ends when all participants reach pattern 53 and drop out one by one.

5 Implications and Future Directions

The concept of a roomful of musicians and "non-musicians" sharing an authentic, although technology-mediated, musical ensemble experience is compelling. This framework was designed to reach a broad general audience, and past participants in public readings have pointed out its potential to bring deep musical experiences to those often left out, e.g. "non-musicians" or persons with disabilities. It is important to note that this is a web app specifically designed to foster music appreciation via in-person interaction and collaboration, ideally in a common space.

While our current implementation is focused on *In C*, the software is general enough to accommodate new compositions and player interactions, simply by defining appropriate JSON data structures to support different musical ideas (not limited to classical music). The current version of our digital music course that inspired this project now features an exercise in which students prepare musical data objects (for sampling, beat patterns, melodic sequences, and Markov chains) that are uploaded to a web app modeled on the one we are presenting here.

6 Acknowledgments

We send our thanks to students in COMP/MUSC 122 Intro to Digital Music at Loyola University Chicago for their many coding and testing contributions to this project. We would also like to thank Access Contemporary Music for supporting the mobile ensemble play-along at their public events.

7 References

[1] Cycling74 Max, 2024. https://cycling74.com/products/max/.

[2] R. Carl. *Terry Riley's In C.* New Music USA, 2010. https://newmusicusa.org/nmbx/terry-rileys-in-c/.

[3] Y. Mann. Tone.js, 2014. https://tonejs.github.io/.

[4] L. McCarthy. P5.js, 2014. https://p5js.org/.

[5] T. Riley. *In C.* Associated Music Publishers, 1964.