Live Coding Ensemble as Accessible Classroom

William Payne
New York University
william.payne@nyu.edu

Matthew Kaney
LiveCode.NYC
Matthew.s.kaney@gmail.com

Yuhua Cao
New York University
yc5884@nyu.edu

Eric Xu
New York University
ehx207@nyu.edu

Xinran Shen
New York University
xs1115@nyu.edu

Katrina Lee
New York University
kal659@nyu.edu

ABSTRACT

In music and computer science classrooms, Blind and Visually Impaired (BVI) learners are often not given alternatives to visual technologies and materials. FiLOrk, an ensemble at the Filomen M. D'Agostino Greenberg Music School, is made up of five BVI high school learners who studied and performed computer music using the live coding language Tidal Cycles over the course of a semester. To make FiLOrk approachable and accessible we wrote a new curriculum featuring audio/tactile learning materials, and we designed a collaborative web editor for use with learners' assistive technologies, including screen readers and braille displays. In this article, we describe findings from classroom observations and interviews. We highlight how learners wrestled with persistent accessibility challenges, connected preexisting music knowledge with Tidal Cycles concepts, created a culture of respect and support, and made suggestions for improving FiLOrk. We conclude by discussing opportunities to make live coding ensembles accessible to both BVI people and high school learners.

Author Keywords

Blindness, Vision Impairment, Accessibility, Live Coding, Laptop Ensemble, Tidal Cycles

CCS Concepts

•**Human-centered computing** → *Empirical studies in accessibility;* •**Applied computing** → **Collaborative learning; Sound and music computing;**

1. INTRODUCTION

Blind and Visually Impaired (BVI) learners are disadvantaged in both computer programming and electronic music because they are highly ocularcentric–i.e. reliant on vision ability to access (Figure 1). Programming environments designed for learning, like Scratch, use visual drag-and-drop interfaces [20, 28], while software development tools use colorful text, underlines, and suggestions to aid sighted programmers [27]. Similarly, commercial music applications use highly visual graphic user interfaces [25, 30].

To navigate visual interfaces, BVI people use assistive technologies like screen readers that verbalize screen contents, or refreshable braille displays [2]. Unfortunately, many software applications do not support assistive technologies. For example, neither Finale, used widely by composers [15], nor Scratch, used widely by young coders, [28] support screen readers.

We formed a collaborative live coding ensemble at The Fil' Community Music School [8] made up entirely of BVI high school learners to address ocularcentrism in music and computing education. We designed text.management, a collaborative coding environment, and we created tactile and large print learning materials. For twelve weeks, the five members of FiLOrk (Fil' Laptop Orchestra, name chosen by members despite the ensemble's size) learned to generate beats and melodies with Tidal Cycles (Tidal for short) [17] and performed live. In this paper, we first list research questions guiding FiLOrk. We then indicate influential prior work. We explain our learning environment and interview protocol. Finally, we share learners' experiences rehearsing and performing, and we suggest future areas of inquiry.

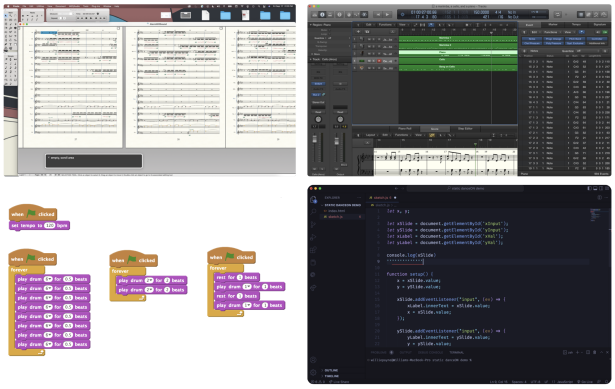

Figure 1: Clockwise from top left - Finale music notation software, Logic Pro digital audio workstation, Scratch blocks-based code environment, VSCode Development Environment. Music and programming software use visual interfaces often inaccessible or limited for BVI users [24, 25].

1.1 Research Questions

Our guiding research question is: How can collaborative live coding be made accessible to novice BVI learners? We aim to address this question through technological, curricular, and collaborative interventions.

NIME'23, 31 May–2 June, 2023, Mexico City, Mexico.

1.1.1 Technology

- What accessibility barriers exist in common tools used by laptop ensembles and live coders?
- How can developers improve accessibility?

1.1.2 Curriculum

- In what ways are current teaching methods ill-suited to BVI learners?
- How can the environment, e.g. topics, instructional materials, and classroom setup, be adapted to support BVI learners?

1.1.3 Collaboration

- How do learners notate and share their ideas?
- How do learners communicate during lessons and performance?

2. RELATED WORK

FiLOrk is primarily influenced by laptop ensemble / orchestra and live coding movements.

2.1 Laptop Ensemble Pedagogy

Laptop ensembles, such as the Princeton Laptop Orchestra (PLOrk), offer an educational model based on collaborative computer music performance [33, 34]. Music provides a practical entry point for novice coders, while the ensemble environment supports individual exploration and collaborative learning. Just as PLOrk was aware of its orchestral lineage, (e.g. using terms like 'conductor' and 'concertmaster' and equipping musicians with speaker arrays that could produce sounds akin to acoustic instruments), we envision FiLOrk as a response to western classical ocularcentrisms like *sight* reading and *watching* the conductor. Prior laptop orchestra research has explored visual notation [11] and communication strategies [7] unsuitable for BVI learners.

2.2 Live Coding and Inclusion

Community norms around open software and inclusion make live coding a fertile ground for accessibility research, though best practices at supporting disabled people have only recently been explored. For example, Skuse interviewed musicians wide a variety of disabilities to incorporate their needs and suggestions in the in future live coding designs [31]. Although no participants explicitly identified as BVI, many suggestions – such as configurable interfaces – are relevant to FiLOrk. Subsequent work explored the technical challenges implementing collaborative music systems accessibly [32].

A common "accessibility" concern of live code performance is the extent to which the underlying technical, algorithmic sound generators are legible to a general audience. Code is frequently projected, so its readability presumes a visual presentation of text and applies theories of image analysis [13]. Prior projects have attempted to make live code more understandable through annotations [29] and novel visualization [19], but have rarely, if ever, extended beyond visual signifiers. Strikingly, the performance *pointilism* (2012) emphasized the opacity of code by rendering it in braille, using it as an "empty symbol," only understandable as abstract symbols [14]. The performance assumes no braille knowledge among the audience, and the fact that images of braille are inaccessible to BVI audience members is an inadvertent (though thematically appropriate) outcome.

2.2.1 Live Coding Education

Many live coding environments have been taught in k-12 classrooms [16, 29, 9, 1, 5, 21] demonstrating significant affordances for both music and computing education. FiLOrk builds on this prior work through highlighting the needs and abilities of BVI learners.

3. METHODS

Broadly, we use a Design Based Research approach in which we situate technical and curricular designs in an authentic learning environment, and we report on how that environment impacts our understanding of the interventions' effectiveness [4]. This study, centered around FiLOrk's inaugural 12-week semester, reflects a nascent phase of our ongoing research. Below, we report how we organized FiLOrk, designed technologies and learning materials, and interviewed participants.

3.1 FiLOrk Overview

Figure 2: FiLOrk's inaugural performance. From left: Mateo (iPad), Cindy (laptop), Olivia (iPad, braille display), Alice (laptop), Donna (iPad).

FiLOrk gathered in a classroom for 45 minutes during an all-day comprehensive learning program on Saturdays. FiLOrk's first semester curriculum combined lessons on Tidal with opportunities to practice ensemble performance (Table 1). We drew from existing resources, e.g. "Learning Tidal" by Alex McLean, but emphasized topics we felt would be appealing and approachable for this group of high school-age participants. For example, we covered Euclidean rhythms, given that learners had taken a drum circle class, and we avoided longer samples because we felt chopping audio could be difficult. We typically introduced two concepts per week.

3.2 Participants

Five high school learners joined. Two identify as blind while three identify as visually impaired. As shown in Table 2, learners range in age and instrument, but all sing and have formal training in western music theory and notation. Participants knew each other prior to joining the ensemble.

3.3 Hardware and Software Setup

Participants used three distinct technology setups. Cindy and Alice (pseudonyms) used a Macbook Pro, VoiceOver screen reader, and braille display. Due to technical issues Cindy did not use her display between weeks nine and eleven while Alice began bringing her display in week three. Mateo and Donna used iPads with text enlargement and external

Wk.	New Topic(s)	Example Syntax	Classroom Style	Absences
1	samples, rest, silence	s “bd ∼sn”	lecture	
2	repeating and replicating patterns	s “bd*4 sn!2”	open work	
3	euclidean rhythms, slowing patterns	s “bd(3, 8) sn/2”	open work	
4	phase, alternating between patterns	s “bd(<3 5>, 8, 1)”	groups	
5	sample numbers	n “0 1 2 3” # s “arpy”	performance practice	
6	changing pattern playback rate	fast 2 $ s “bd sn”	groups	
7	layering patterns	n “0 1” # s “<bd sn>”	groups	
8	randomness, effects with 0–1 arguments	s “bd*8?” # pan 0.2	open work	
9	effects with non-normal arguments	s “sn*5” # vowel “a e i o u”	performance practice	Alice, Mateo
10	oscillators	s “bd*8” # pan sine	lecture	
11	composition planning		open work	Alice, Donna
12	rehearsal, concatenating patterns	cat [s “bd bd” , s “sn sn”]	performance practice	
13	performance			

Table 1: FiLOrk Class Structure

```
62  -- Donna
63  donna $ cat [n "[1 2] [3 4] [5 1] [2 3]",
64   n "[1 1] [6 1] 1 1",
65   n "[6 1] 6 [1 1] [6 6]",
66   n "[1 ~ 1 1] [1 1] 1 1"]
67  # sound "sn" -- Apply randomness, then multiply
68  # room "0" -- Between 0 and 1 (try 0.7)
69  # vowel "i" -- a e i o u
70  # gain "1" -- Between 0 and 1
71  # delay "0" -- Between 0 and 1 THEN CHANGE TO ARPY
72  # squiz "0" -- Even numbers
73  # crush "20" -- Standard is 20
74
75  donna silence
76
77  ------------------------------------------------------------------
78
79  -- Setup
80
81  setcps(125/60/4)
```

Figure 3: Donna’s performance code is shown in the text.management online live-coding editor. The pattern in lines 63–66 is held constant, while learners adjusted the sounds and effects as the performance progressed.

keyboards. Olivia, a learner with braille knowledge and some vision ability, used a hybrid of her peers’ approaches consisting of an iPad, VoiceOver, and Mantis, a combination braille display and QWERTY keyboard.

In class, learners coded in text.management, a browser-based collaborative live-coding environment we designed to support screen readers and meet needs as they arose. Detailed further in [12], text.management uses CodeMirror 6, a recent version of the popular code editor library with improved screen reader support [10], which we have been configuring to mitigate VoiceOver bugs, such as with line highlight and the default keybindings. All music was executed by Tidal running on the lead researcher’s computer. VoiceOver users heard their screen readers only on their devices. To support home use, we installed Tidal and VSCode for the two laptop users, and we demonstrated Estuary, a web-based live coding environment that supports a subset of Tidal [22].

3.4 Learning Materials

As learners cannot read projected code, we demonstrated directly in text.management and created supplemental materials:

1. The **class website** was designed to be accessed with screen readers and used outside of class.[1]
2. Weekly **reference handouts** in large print and braille were made to present brief summaries of concepts with code examples as learners cannot easily access multiple windows on screen in parallel.
3. **Tactile graphics** were made using braille and swell touch, a special paper that makes ink rise when heated through a swell form machine. We adapted two concepts conventionally taught visually: Euclidean rhythms, represented with geometric shapes superimposed over a circle, and oscillators, represented as amplitude curves graphed over time. We printed a standard ink layer making them universally accessible (Figure 4).

[1]FiLOrk class website started in 2022: https://huriphoonado.github.io/laptopclass/

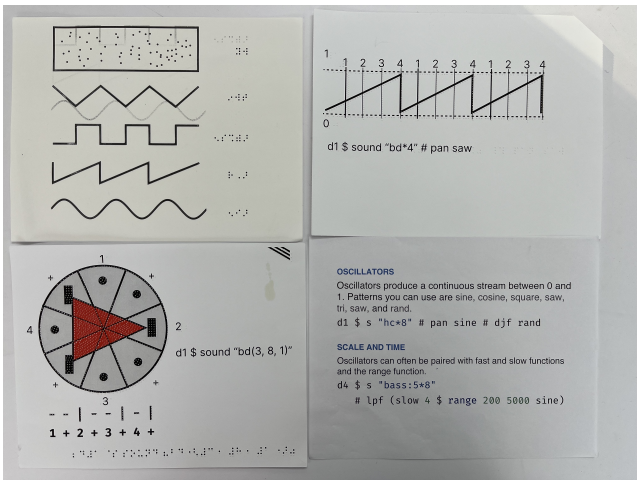

Figure 4: Tactile graphics used to teach waveforms and Euclidean rhythms containing swellform, braille, and print, and a large print reference handout (bottom right).

3.5 Classroom Styles

Classes were run by the lead researcher and one or two assistants. Each week consisted of technology setup, recap, and main activity. We explored four primary teaching styles:

1. **Lecture**: We presented new concepts and prompted learner responses.
2. **Open Work**: Learners sat around a large table. Each worked independently in a commented section of the document. Generally, learners worked with similar code using different sounds.

3. **Groups**: We split the class into two groups, (Cindy / Olivia / Mateo and Donna / Alice), at separate tables. Each group was provided starter code, instructions, and their audio was separated between monitors.
4. **Performance Practice**: We composed a short piece involving starter code and instructions for learners to make changes over time, e.g. modifying rhythms, applying effects, etc. Performance practices were intended to help learners practice live coding, listening, and signalling.

3.6 Interview Protocol

After a final performance, we conducted five semi-structured, hour-long Zoom interviews that were recorded and transcribed verbatim. Interviews were led by the first author, while an additional researcher took notes and asked follow-up questions. First, in a warm-up, we asked learners to compare expectations with their experiences. Then, we asked demographic questions to gather self-reported pronouns, vision descriptions, and prior experience. What followed was guided by our three research questions (§1.1): We asked participants to discuss using text.management with assistive technologies, learning Tidal, and collaborating. We derived specific questions from classroom observations–for example, we asked Donna to share her process leading the final performance, and we asked Olivia to describe how she used VoiceOver. Finally, we asked learners to share hopes and suggestions for the ensemble's future.

3.6.1 Analysis

We conducted a thematic analysis on the interview transcripts resembling Braun and Clarke's method [3]. We used a combination of deductive and inductive coding. While we began with three overarching categories guided by our research questions–Technology, Curriculum, Collaboration–codes emerged out of interviews. Following each interview, two researchers present discussed and listed initial codes to make sense of what they heard. After interviews concluded, the research team worked independently to highlight quotations, tag with existing codes, and/or suggest new codes. We then met to discuss the codebook and identify redundancies and outliers (i.e. rarely used codes). After one more iteration, in which researchers reviewed transcripts and noted quotations with disagreement, we met to achieve consensus. In preparing this document, we refined themes and selected especially pertinent quotations.

4. FINDINGS

Our findings are organized by our research questions (§1.1): technological, curricular, and collaborative components of making a laptop ensemble accessible to BVI learners.

4.1 Technology

text.management enabled learners to write Tidal code using a range of assistive technologies and hardware setups. They described how they accessed text.management, typed code, and used Tidal at home.

4.1.1 Accessibility of text.management

Learners found text.management accessible compared with previous experiences. Cindy told us, "Code Academy didn't work with the screen reader I was using. So, my sister read everything, and I would dictate what I wanted, and then she would type it. It was very frustrating. That's why I quit." Donna similarly reflected, "I had to get a laptop with Zoom Text so I could actually see my code because it wasn't like Safari or something. It was awful. At the start of every class I would spend half of it fighting with my laptop. Because the laptop was outdated it ran slower, and it was running more programs because of Zoom Text."

text.management worked with learners' preferred setups. Both Donna and Mateo used iPads due to the ease of navigation afforded by the touchscreen. For example, Mateo said, "I prefer pinching because it's easier to maneuver around," while Donna said, "I did some combination of zooming by pinching and also the triple tap. I did have it zoomed-in a good amount, where if I had a long line of code, like our final project, I wouldn't have seen it all without scrolling, which is why I put each measure on a separate line." Cindy and Alice used text.management with and without braille displays, finding they improved navigation. Alice said, "it was a little difficult not having my braille display. My braille display made it a lot easier." Cindy suggested, "when it works, braille revolutionizes things. I should have done this earlier, but using headphones for computer speech helps a lot. You have to turn the volume up, but at least you have your own isolated audio you can make talk as verbosely as you want without interrupting anyone else."

Olivia faced significant accessibility barriers with her unique setup, finding that VoiceOver on iPad misrepresented code: "It was separating whenever there was a space. I'd have to scroll over each individual part instead of reading the entire line. Then when I try to click, it selects the entire line." As a result, Olivia often held the iPad close to her face and manually positioned her cursor through touch: "I had to keep turning VoiceOver off and on again to get my cursor right." While Olivia said "I can deal with it," it was "frustrating at times."

4.1.2 Typing Music

Learners contrasted their live coding experiences with other music technologies, citing possibilities and challenges afforded by the open-ended text editor. Cindy said, "I think the interface is simpler. You're writing everything so you have ultimate control over your environment. But, it's also more complicated: You can't just hit a button and change a parameter or drag a slider. You have to remember all the syntax, and you have to write it correctly." Similarly, Mateo felt that "some flaws of having such an open ended canvas is finding a direction – Say you want to make a beat or something and you know what it is, you just don't know how to notate it."

Learners described how typing could be mechanically difficult. For example, Olivia said, "I'm stopping and checking every few characters. I keep forgetting which is the right key: which is slash, which is dollar sign, and the angle brackets are a bit frustrating to find. Like code: I've just been learning for four months, but I've been typing for over ten years." text.management, like other editors, auto-completes enclosures such as quotation marks and parentheses to save keystrokes, but learners disliked the feature. Donna said "if I type my closed quote, it gives me the open and the closed, and I have to delete the open. I think it's weird when you get more than you typed." Furthermore, VoiceOver did not announce additional enclosure insertions, meaning Cindy, Alice, and Olivia could be unaware of visible characters.

4.1.3 Limited Home Access

Name	Pronouns	Vision Ability	Grade	Hardware / Assistive Technology	Prior Code Experience	Primary Instrument(s)
Mateo	He/Him	Visually Impaired	9	iPad, keyboard / Magnification	None	Voice Piano, Guitar
Donna	She/Her	Visually Impaired	11	iPad, keyboard / Magnification	High school class	Voice Electric Bass
Olivia	She/Her	Visually Impaired	11	iPad, Mantis keyboard / VoiceOver, braille display	None	Voice Piano
Cindy	She/Her	Blind	11	MacBook Pro / VoiceOver, braille display	Independent	Voice Piano, Violin
Alice	She/Her	Blind	12	MacBook Pro / VoiceOver, braille display	High school class	Voice Piano

Table 2: FiLOrk Members

Olivia, Donna, and Cindy experienced difficulties using Tidal outside of text.management. (Mateo and Alice did not try.) Olivia and Donna used Estuary [22]. Olivia reflected that, "I couldn't make any sound, and the code there is different and I didn't want to confuse myself more," while Donna similarly remembered, "I would put in a rhythm I knew, and it wouldn't come out right. I think that's the iPad because I tried Estuary on my friend's laptop and it worked fine." Cindy used Tidal with VSCode, experiencing unexpected lag: "I was having fun when it worked. I did notice Visual Studio, or the combination of Visual Studio, SuperCollider and VoiceOver, oftentimes lagged. I don't know if it's my computer, or the programs, or a combination. It was probably because it made noises on my computer itself rather than sending them to yours during class. It was a lot laggier." Furthermore, Cindy's installation broke after an update, needing help to reinstall Haskell. Finally, while Mateo did not write code outside class, he took photos to show his family,"this is what I learned, like how and why I am using it."

4.1.4 *Technology Recommendations*

VoiceOver users requested that we fix how text.management conveys text. (As discussed below in Section 4.3, collaborative editing exacerbates bugs.) Most also felt that error feedback, shown in an adjacent area, could be improved. Donna pointed out that "everybody's errors get put there," instead of it highlighting "Olivia has an error, Mateo has an error," so it would be better if it could "denote that you have triggered your code and there's something wrong."

4.2 Curriculum

Learners found the curriculum relevant, enjoyable, and accessible. For example, Donna noted that the "diagrams for Euclidean rhythms and waveforms are really good because you could not have explained a saw wave if you tried verbally," while Cindy felt that the "website was good because there wasn't too much we had to do to start coding. Everything was legible." However, learners discussed challenges remembering syntax, reconciling their rhythm understanding with cycles, and understanding code as it grew complex.

4.2.1 *Reading & Remembering Syntax*

Many difficulties arose out of Tidal syntax and behaviors. For example, Olivia argued that effects arguments felt arbitrary: "I have trouble remembering - like what is from 0 to 1, what is even numbers, why does this only go up to 10 or something?" Cindy told us the difficulty remembering syntax rules is complicated by VoiceOver limitations: "The little things become difficult. If you put something using pipe, you put it in brackets. There are other examples... Like slow and fast, you put in parentheses outside the quotes. It's especially difficult for screen readers because they don't announce this by default. So you just have to remember and hope it works." Both Mateo and Donna felt that while the tactile graphics helped them understand the concept of Euclidean rhythms, it did not help them remember the syntax (Table 1). Mateo said, "it was a lot easier to to visualize what was going to happen than Tidal on the page."

4.2.2 *Reconciling Rhythms & Meters with Cycles*

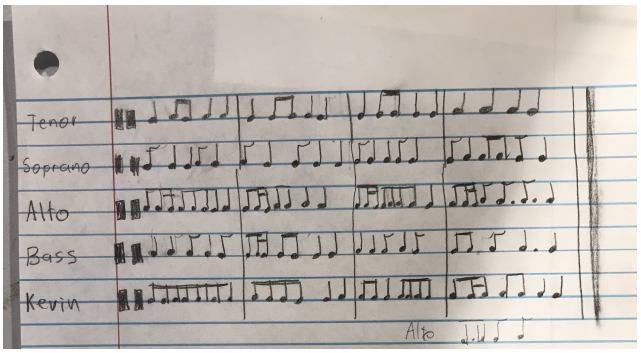

Figure 5: Rhythms hand-written by Donna and later converted to Tidal patterns for the performance. The alto and bass parts were especially difficult due to dotted eighth and quarter notes. (The part labelled "Kevin" is an inside-joke.)

All were experienced with western music conventions and relied on prior rhythmic knowledge when attempting to conceive of and code cyclical patterns. At times, prior knowledge aided code comprehension. Donna argued, "because we all have a solid foundation, we could hear our output and know what had gone wrong in terms of, 'well, I wanted eighth eighth quarter eighth eighth quarter' and I got 'quarter quarter quarter quarter.'" Cindy similarly felt, "generating sounds within a 4/4 pattern, and grouping things by bracket, and using asterisks and stuff was pretty easy because I could fit it into a 4/4 grid in my head." At other times, learners grew frustrated when they were unable to code a rhythm they knew. In the final project, Donna transcribed five rhythms for each of the learners to code (Figure 5). Two rhythms with dotted notes were difficult, and Olivia remembered, "the dotted eighth note that I just couldn't – that I had some help figuring out how to write. For quarter notes, I can write like one underscore. Is that right? But there's no dotted eighth."

4.2.3 *Debugging Randomness and Effects*

Learners relied on their ears to discern whether code executed and behaved as expected. However, as Donna summarized, "you don't know exactly what your code is going to sound like when it gets to a certain level of complexity." Learners identified how randomness and some effects made changes imperceptible. Olivia said "I think the most frustrating for me was randomness. Pipe. That confused me so much. Every beat has whatever chance to be played." Donna identified perception limitations during performance, "there was so much going on, and especially with randomness you can't tell if it was hitting a new cycle or if it was the change you wanted. Sometimes with delay and randomness and squiz [distortion] and stuff, you don't know what it's going to sound like, so you might not be paying attention for the right change." Mateo considered, "the performance aspect is really fun because you never know what is going to happen, but that is also the stressful thing."

4.2.4 *Curriculum Recommendations*

While participants appreciated the learning environment, they offered suggestions. Cindy and Donna felt we could improve the Euclidean rhythm tactile graphics for sighted and blind learners. Donna suggested utilizing color to map graphic elements to code: "The three values in the Euclidean rhythm - number of beats per divisions of a cycle with x delay - if you put those in corresponding colors, it might have been faster to click - 'Oh, this value does that.'" Cindy suggested we consult an expert, stating she felt like our tactile graphics were "created from the perspective of a sighted person," and though "tactile graphics are good... having the dual perspective of blind people who have gone through STEM in college would be useful." Additionally, Donna emphasized teaching how to recognize common Tidal syntactic patterns: "Instead of say, introducing gain as a thing, and then squiz as another thing, do: 'space after your quotes, hashtag, the effect, and then a value.'" Finally, all felt that instructors offered help too quickly. Olivia requested, "you explain the basics and then let us loose."

4.3 Collaboration

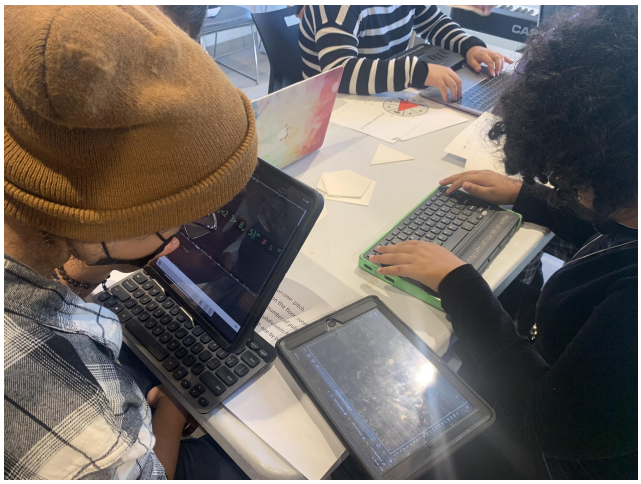

Figure 6: Learners in rehearsal. iPads, tactile graphics, and keyboards are scattered around.

Learners enjoyed collaborating. Alice summarized, "we actually stood together, and we listened to each other." They detailed virtual collaboration inside text.management and strategies to maintain awareness and respect in the classroom.

4.3.1 *Lack of Screen Reader Support for Co-Editing*

A major technical difficulty we discovered is that VoiceOver does not trigger audio feedback in Safari when other users make edits. While Mateo and Donna were able to see nearby changes with magnified text, Cindy, Alice, and Olivia did not know when someone else edited code on the same line until they made a change or moved their cursor. As a result, the three VoiceOver users found group activities disorienting. Cindy said, "Mateo and I were working on the same code, and it was really hard to keep up because braille doesn't refresh automatically. When there were syntactical mistakes, I couldn't read everything fast enough to determine the problem. So, I ended up getting really frustrated because everything was out of my control." Olivia felt, "it was easiest when we worked on separate lines," but the unified text area meant learners could still accidentally disrupt others. Near the start of the class, Alice held option while moving her cursor to navigate, consequently shuffling everyone's code. (By default, VoiceOver commands use command and option keys held together.)

4.3.2 *Collaborative Live Coding Versus Writing*

Learners identified practical differences between live coding and collaborating in Google Docs. Donna felt live coding is more independent: "I care more about what somebody in a Google Doc is typing than what exactly Olivia's code is - unless she has a problem that she needs me to look at. Even though everything in text.management happens at the same time, it's very much independent. If you're editing a history project, and you know that somebody didn't explain Dag Hammarskjöld, you now need to explain before you talk about the fact that he is suddenly dead. There's more referring back and forth in Google Docs." Alice noted that Google Docs do not produce audio output unlike live coding where she "could actually hear what the other person is typing."

4.3.3 *Importance of Support and Respect*

All learners suggested FiLOrk was more open-ended than other ensembles they perform in together. Mateo said, "In vocal ensemble, you're learning parts. Everything's very planned. Everything's cohesive. In [FiLOrk] there are no sections, there's not really a part system, and it's free how you go about doing things." Without assigned music or structure, learners had to reconcile contrasting aesthetics. For example, Olivia said, "each of us has our own style. Cindy likes her crazy rhythms and her evil plans. I usually stick to normal stuff, but I like a sprinkle of weirdness, a little surprise." Learners could impose their style on others through editing or more often silencing other channels or the full ensemble. Donna remembered, "I've been known to silence Cindy, but that was mostly in the beginning when things were horrendous." Learners believed that context was important for determining when to silence other channels. While Cindy felt "we weren't as adventurous as maybe we could have been," "nobody really came at each other like 'Oh, turn off your audio because it sucks.' That was never really the thing." Instead, learners asked permission when silencing and usually did so to manage classroom noise.

Learners looked out for each other and offered help, especially with technical difficulties. In particular, multiple

learners highlighted how Cindy repeatedly aided Alice, who was less adept at VoiceOver. Alice remembered "it was Cindy when we were trying to fix something that had to do with VoiceOver or my laptop," while Donna commented, "If there's a problem with VoiceOver, Cindy's got it."

4.3.4 Collaboration Recommendations

Mateo and Donna suggested we incorporate color and audio feedback to indicate a text fragment's source. Cindy requested options for VoiceOver to "not read collaborators' actions on the document" but "announce who is editing." To improve peer support in the classroom, Donna suggested we pair learners with similar assistive technologies rather than split them up: "If Alice has a problem, not only is she using VoiceOver, which I'm not fluent in, she's also on a laptop, and I don't know what to do with a laptop. Cindy and Alice are using the same tech setup, so have the laptop people and the tablet people working together. It's easier for Olivia to turn her screen on than it is for me and Mateo to read Cindy and Alice's laptops." Finally, Donna hoped that the ensemble will explore other strategies than speaking to each other during performance: Donna said, "I was lobbying for non-spoken signals. I would love it if nobody talks, there's very little sound coming from humans, and all the sound is from computers."

5. DISCUSSION

In general, FiLOrk met our initial technology, curriculum, and collaboration goals because all learners enjoyed their experience and successfully performed. Yet there is room for improvements. Below, we offer technical design suggestions to make the live coding ecosystem more accessible to BVI people, describe opportunities and challenges pertaining to teaching live coding, and conclude with suggestions for making other live coding ensembles accessible and inclusive for diverse learners.

5.1 Tech Trade-offs For BVI Live Coders

Despite technical challenges, text.management facilitated live coding in the classroom because it provided flexible use options: Visually impaired learners chose tablets to aid navigation while screen reader users accessed code through headphones or braille while listening to Tidal output in the room (§4.1.1). Other environments did not support learners at home because they performed poorly on tablets or were not optimized to support screen readers (§4.1.3). In general, the live coding ecosystem requires installing software, but installation is not only complicated for novices, development environments can be complex, or in the case of Atom, inaccessible[2]. Browser-based solutions, in contrast, support a range of devices, but may limit developers' ability to fine-tune text-to-speech output, as shown by text.management's strange text bugs (§4.1.2). While researchers have explored audio/tactile feedback for screen reader users in text-based composition [23], there are opportunities to further explore screen reader interactions in live-coding that balance awareness and music.

5.2 Syntax for Accessible Teaching

Tidal is a concise syntax designed to be typed and edited quickly during performances [17]. Its concision simplifies navigation. Low vision learners saw most of their code while zoomed in, while blind learners fit much of their code on their braille displays despite being limited to 40 cells (§4.1.1). As with prior research [23], our learners perceived inputting music with text to be sluggish compared to playing on a MIDI keyboard. However, in this learning context, we feel that developing keyboard literacy is an added benefit.

An unforeseen consequence of Tidal's concision is that learners felt it was hard to remember syntax and the purpose of function arguments. For example, learners eventually comprehended Euclidean rhythms, but they they faced difficulty understanding and remembering syntax, e.g. the order of three numbers and the type of separation. One solution may be to support optional labelled arguments, e.g. `bd(3, 8, 2)` could be written as `bd(pulse:3, step:8, offset:2)`. Existing IDE and Tidal-specific tools to aid remembering syntax and understanding behavior are visual, e.g. [19, 18]. Given that auto-completed enclosures were not detected by VoiceOver in browsers (§4.1.2), other visual cues, like code suggestions, are unlikely to be accessible without additional development.

5.3 Collaboration Design Opportunities

The most significant barrier experienced by VoiceOver users was the lack of feedback when others edited their code (§4.3.1). Solutions require nuance. If VoiceOver announced every change, the amount of feedback would be overwhelming, and participants felt they did not need to know others' precise actions. Still, as collaborative live coding performances have shown, we believe there is significant creative and educational potential for BVI collaborators to make real-time changes to each others' code. Live coding presents a unique environment to understand accessible collaboration given its strict timing requirements and loud environment. Other researchers have explored accessible collaborative document editing [6] and software development through tools such as CodeWalk, designed to support mixed-vision pair-programming with audio cues to convey navigation/edit actions and unique commands to tether screen reader output to a leader [26]. To support collaborative live coding, a combination of universal cues (e.g. when two people enter the same line) and distinct modes (e.g. solo editing vs. listening) may achieve the best results across circumstances. Design findings may impact mixed-ability interactions in other synchronous environments.

5.4 Aesthetic Differences as Opportunities

We focused on designing software and teaching materials, but we discovered the importance of interpersonal relationships in facilitating a successful class. In FiLOrk, learners knew each other and were aware of each other's strengths and weaknesses. They felt comfortable offering and asking for help (§4.3.3). However, they acknowledged that certain actions could appear disrespectful. We suggest that other high school live coding ensembles discuss ground rules up front and codify requirements for polite behavior, e.g. asking permission before silencing another channel.

Furthermore, while our performance practices required that learners stay synchronized while editing similar code structures (e.g. changing the same effects, moving down one line, etc.), we intend to challenge learners to explore asymmetric layouts and to treat aesthetic differences as creative opportunities. For example Cindy's distinct love of atonal, arrhythmic music may create tension (§4.3.3). We believe it suggests an improvisation in which Cindy experiments while others maintain stability, demonstrating to learners

[2]Atom does not support screen readers: `https://github.com/atom/atom/issues/18660`

that their unique group dynamic may yield surprising outcomes.

6. LIMITATIONS AND FUTURE WORK

While FiLOrk represents a diverse range of vision ability, musical skill, and technical literacy, it is small, and all five learners knew each other. In the future, we intend to continue teaching Tidal concepts with tactile learning materials and encourage learners to work on larger, more complex compositions.

7. CONCLUSION

We presented FiLOrk, a live coding ensemble made up of five BVI learners. We successfully taught for twelve weeks using accessible course materials and text.management, a browser-based editor designed to work across platforms and assistive technologies. Learners explained how they utilized text.management and faced challenges with VoiceOver, how they enjoyed Tidal but felt challenged by syntax and certain rhythms, and finally how they worked together within the document and classroom. Ahead, we believe there are opportunities to improve live coding accessibility and inclusion for BVI learners through technological, curricular, and collaborative interventions.

8. ACKNOWLEDGEMENTS

We thank the FMDG School faculty and students for their support and insight. We also thank members of the NYU Ability Project and Vertically Integrated Projects, especially Stella Yu.

9. ETHICAL STANDARDS

This research is approved by New York University's Institutional Review Board. All participants are minors who signed assent forms while their parents signed consent forms and were given ample time before, during, and after interviews to opt out of research. We shared forms in text, Microsoft Word, and PDF formats to ensure they could be accessed across assistive technology preferences. Participants were compensated $25 for signing up for interviews.

To ensure the establishment of shared goals, the lead author has volunteered at The Filomen M. D'Agostino Greenberg Music School for three years prior to this study. He has developed a relationship with the learners through many non-research activities including helping lead other ensembles, creating braille and large print music, and assisting with performances. FiLOrk meets the needs and interests of multiple parties beyond our team. It originated out of conversations between FMDG Music School teachers who wanted an electronic music ensemble, learners who wanted more technology offerings, and members of our research team who practice live coding. Not only did participants elect to enroll in the first semester of FiLOrk depicted in this paper, all five chose to enroll in a second semester.

10. ADDITIONAL AUTHORS

Amy Hurst, email: amyhurst@nyu.edu.

11. REFERENCES

[1] S. Aaron, A. F. Blackwell, and P. Burnard. The development of sonic pi and its use in educational partnerships: Co-creating pedagogies for learning computer programming. *Journal of Music, Technology & Education*, 9(1):75–94, 2016.

[2] American Federation of the Blind. Screen readers, 2022.

[3] V. Braun and V. Clarke. Using thematic analysis in psychology. *Qualitative Research in Psychology*, 3(2):77–101, 1 2006.

[4] D.-B. R. Collective. Design-based research: An emerging paradigm for educational inquiry. *Educational Researcher*, 32(1):5–8, 2003.

[5] N. Collins. Live coding and teaching SuperCollider. *Journal of Music, Technology and Education*, 9(1):5–16, May 2016.

[6] M. Das, A. M. Piper, and D. Gergle. Design and evaluation of accessible collaborative writing techniques for people with vision impairments. *ACM Trans. Comput.-Hum. Interact.*, 29(2), jan 2022.

[7] T. Edwards and R. B. Sutherland. Eyes off the screen! techniques for restoring visual freedom in leo performance. In *Proceedings of the 1st Symposium on Laptop Ensembles & Orchestras (SLEO)*, pages 33–40, 4 2012.

[8] Filomen M. D'Agostino Greenberg Music School. Fmdg school: Fostering education, access, and inclusion for people of all ages with vision loss, 2021.

[9] J. Freeman and B. Magerko. Iterative composition, coding and pedagogy: A case study in live coding with EarSketch. *Journal of Music, Technology and Education*, 9(1):57–74, May 2016.

[10] M. Haverbeke. Screen reader demo, 2022.

[11] S. Hewitt and P. A. Tremblay. Notational approaches for laptop ensembles. In *Proceedings of the Symposium on Laptop Ensembles and Orchestras*, pages 72–75, 4 2012.

[12] M. Kaney, W. Payne, and A. Hurst. Addressing accessibility for blind and visually impaired live coders. In *Proceedings of the 7th International Conference on Live Coding (ICLC '23)*, Utrecht, The Netherlands, 4 2023. TopLap.

[13] S. Lawson and R. R. Smith. What Am I Looking At?: An Approach to Describing the Projected Image in Live-Coding Performance. In *Proceedings of the Fourth International Conference on Live Coding*, page 353, Madrid, Spain, Jan. 2019. Medialab Prado / Madrid Destino.

[14] I. m. zmölnig. Audience perception of code. *International Journal of Performance Arts and Digital Media*, 12(2):207–212, July 2016.

[15] makemusic. finale music notation software, 2021.

[16] B. Manaris, B. Stevens, and A. R. Brown. Jythonmusic: An environment for teaching algorithmic music composition, dynamic coding and musical performativity. 9(1):33–56, May 2016.

[17] A. McLean. Making programming languages to dance to: Live coding with tidal. In *Proceedings of the 2nd ACM SIGPLAN International Workshop on Functional Art, Music, Modeling & Design*, FARM '14, page 63–70, New York, NY, USA, 2014. Association for Computing Machinery.

[18] A. Mclean. Algorithmic pattern. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 265–270. Zenodo, June 2020.

[19] A. McLean, D. Griffiths, N. Collins, and G. Wiggins. Visualisation of live code. In *Proceedings of the 2010 International Conference on Electronic Visualisation*

and the Arts, EVA'10, page 26–30, Swindon, GBR, 2010. BCS Learning & Development Ltd.

[20] L. R. Milne and R. E. Ladner. *Blocks4All: Overcoming Accessibility Barriers to Blocks Programming for Children with Visual Impairments*, page 1–10. Association for Computing Machinery, New York, NY, USA, 2018.

[21] D. Ogborn. Live coding together: Three potentials of collective live coding. *Journal of Music, Technology and Education*, 9(1):17–31, May 2016.

[22] D. Ogborn, J. Beverley, L. N. del Angel, E. Tsabary, A. McLean, and E. Betancur. Estuary: Browser-based collaborative projectional live coding of musical patterns. In *Proceedings of the Second International Conference on Live Coding*, 2017.

[23] W. Payne, F. Ahmed, M. Gardell, R. L. DuBois, and A. Hurst. Soundcells: Designing a browser-based music technology for braille and print notation. In *19th Web for All Conference*, W4A '22, New York, NY, USA, 5 2022. Association for Computing Machinery.

[24] W. Payne and S. A. Ruthmann. Music Making in Scratch: High Floors, Low Ceilings, and Narrow Walls? *Journal of Interactive Technology and Pedagogy*, (15), 2019.

[25] W. C. Payne, A. Y. Xu, F. Ahmed, L. Ye, and A. Hurst. How blind and visually impaired composers, producers, and songwriters leverage and adapt music technology. In *The 22nd International ACM SIGACCESS Conference on Computers and Accessibility*, pages 1–12, 2020.

[26] V. Potluri, M. Pandey, A. Begel, M. Barnett, and S. Reitherman. Codewalk: Facilitating shared awareness in mixed-ability collaborative software development. In *Proceedings of the 24th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '22, New York, NY, USA, 2022. Association for Computing Machinery.

[27] V. Potluri, P. Vaithilingam, S. Iyengar, Y. Vidya, M. Swaminathan, and G. Srinivasa. Codetalk: Improving programming environment accessibility for visually impaired developers. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, page 1–11, New York, NY, USA, 2018. Association for Computing Machinery.

[28] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, et al. Scratch: programming for all. *Communications of the ACM*, 52(11):60–67, 2009.

[29] C. Roberts. Code as information and code as spectacle. *International Journal of Performance Arts and Digital Media*, 12(2):201–206, July 2016.

[30] A. Saha and A. M. Piper. Understanding audio production practices of people with vision impairments. In *The 22nd International ACM SIGACCESS Conference on Computers and Accessibility*, pages 1–13, 2020.

[31] A. Skuse. Disabled Approaches to LiveCoding, Cripping the Code. In *Proceedings of the 2020 International Conference on Live Coding (ICLC2020)*, pages 69–77, Limerick, Ireland, Feb. 2020. University of Limerick.

[32] A. H. C. Skuse and S. Knotts. Creating an online ensemble for home based disabled musicians: Disabled access and universal design - why disabled people must be at the heart of developing technology. In R. Michon and F. Schroeder, editors, *Proceedings of the International Conference on New Interfaces for Musical Expression*, pages 115–120, Birmingham, UK, July 2020. Birmingham City University.

[33] D. Trueman. Why a laptop orchestra? *Organised Sound*, 12(2):171–179, 2007.

[34] G. Wang, D. Trueman, S. Smallwood, and P. R. Cook. The laptop orchestra as classroom. *Computer Music Journal*, 32(1):26–37, 2008.