EVALUATING COLLABORATIVE LAPTOP IMPROVISATION WITH LOLC

Sang Won Lee
Georgia Tech
Center for Music Technology

Jason Freeman
Georgia Tech
Center for Music Technology

Andrew Colella
Georgia Tech
Center for Music Technology

Shannon Yao
Georgia Tech
Digital Media

Akito Van Troyer
MIT
Media Lab

ABSTRACT

This paper discusses LOLC, a text-based collaborative music improvisation system for laptop ensemble developed by the authors. The system is contextualized in terms of related work and the specific motivations and goals for the project, and its design and implementation are explained. The paper then evaluates LOLC in the context of a recent performance by professional classical musicians with minimal computer experience. Using qualitative data from interviews with the performers and quantitative data from server logs, the paper considers the degree to which LOLC facilitated collaborative improvisation among the musicians, and the ways in which using LOLC differed from more traditional modes of collaborative instrumental musical improvisation.

1. INTRODUCTION

Textual performance interfaces can offer a unique environment for collaborative musical improvisation in laptop-based musical ensembles. They can facilitate efficient, flexible communication among ensemble members by supplementing traditional channels of aural and visual communication among members with dialog across digital networks that includes data sharing, time synchronization, and chatting. Such networked communication can also be persistent, enabling musicians to trace back through communication rather than relying solely on their memory of how a performance has transpired.

While many live coding languages do facilitate such text-based interaction across an ensemble [5,3], most are ill-suited for larger laptop ensembles and few correspond to improvisational modes in more traditional ensembles.

We designed and developed LOLC [8] to take advantage of the unique potential of text-based performance environments in larger-ensemble collaborative improvisation and to build upon the rich history of collaborative improvisation in jazz and avant-garde musical styles. In LOLC, musical patterns are coded symbolically and shared automatically, providing a foundation through which laptop musicians can effectively improvise and collaborate by borrowing and transforming the material created by others in the ensemble.

We also wanted to make LOLC accessible to non-programmers, including skilled musicians without any experience in programming or computer music. LOLC is thus deliberately limited in complexity and scope: it is not a Turing-complete programming language and consists entirely of single-line expressions.

In this article, we review the related work upon which LOLC builds and we outline the goals, design and implementation of the system. We then evaluate LOLC in the context of a recent performance by an ensemble of professional classical musicians who had no background in computer programming and little or no background in computer music. We consider the degree to which LOLC facilitated collaborative improvisation among the musicians, the degree to which LOLC was accessible to non-programmers to learn, and the ways in which using LOLC differed from more traditional modes of collaborative instrumental musical improvisation.

2. RELATED WORKS

The design and implementation of LOLC was influenced by existing models for collaborative text-based laptop performance and by approaches to collaborative improvisation in other types of ensembles.

Laptop-based musical ensembles can often collaborate more effectively when they share a common clock and code and/or music over the network. Several live-

coding environments implement such collaboration features over a local–area network. Rohrhuber's JITLib [5] and Sorensen's Impromptu [3], for example, enable clients to share and manipulate dynamic objects or variables over a network.

Instead of sharing dynamic objects or variables, some researchers have suggested an environment that enables users to share actual code fragments among members of the ensemble. A design document for the Co-Audicle [21] supports such exchange with a client-server or a peer-to-peer model. JITLib [5] implements text chat functionality and enables musicians to interpret code directly on each other's machines [18].

Another notable approach in collaborative laptop performance is that many ensembles do not use standardized tools among all performers. Instead, they simply define a shared protocol for communication. The Hub [4], for example, defined a new protocol for the ensemble with each piece they developed, but each musician used their own software to perform it.

The Hub's work Borrowing and Stealing offers an intriguing model that served as a direct inspiration for LOLC. Instead of sharing code or variables, members share music. In the piece, a shared data store maintains symbolic representations of musical fragments created by each player. Musicians then retrieve the fragments created by other players, manipulate them, play them and store the transformed version in the database. LOLC follows this approach upon which musicians can more easily borrow and recreate the materials of other ensemble members.

In addition to the ideas of live coding languages, LOLC was influenced by collaborative improvisation in other types of ensembles. Many composers have created structured strategies for ensemble improvisation in their works. Some of the works include gestures to give players instructions on how to improvise [22]. In Virtual Concerto [11], sections of the orchestra improvise together, following a matrix of instructions. Similar approaches have been proposed in ensemble-based improvisation pedagogy; a "dominoes" exercise, for example, asks players to sit in a circle and play in sequence, closely imitating the gesture of the person preceding them [1].

Many ethnomusicologists have characterized group interaction among improvising jazz musicians as a conversation: "the exchange of the [musical] idea not only established an abstract succession of sounds and rhythms but linked [the musicians] as musical personalities … at a particular moment in time" [15]. Berliner describes how musicians respond to each other, particularly when trading short improvised phrases, noting how "musicians pursue a middle ground that satisfies their desire for both continuity and change by borrowing material from one another and transforming it" [2].

3. DESIGN AND IMPLMENTATION OF LOLC

3.1.Goals and Design Principles

The main goal of LOLC was to create a text-based performance environment to facilitate collaborative improvisation in a laptop ensemble by sharing all musical materials. Furthermore, we wanted LOLC to be readily accessible to novice programmers and even non-programmers. In connection with this second goal, LOLC is not intended to be a full-featured computer music language like ChucK [20] or SuperCollider [13] or even necessarily a language at all. For instance, it uses pre-recorded sound files as musical building blocks instead of supporting sound synthesis or signal processing.

To facilitate accessibility, we designed interaction among musicians to focus on sharing musical content rather than computational content. All pattern definitions are based on the symbolic representation of rhythmic, dynamic, and sound-source information. This makes it easier for musicians to re-use and transform the material they hear others playing. In addition, all LOLC expressions are a single line in length, and there are only two expression types other than chat messages: pattern definitions and scheduling operations.

Like many collaborative performance tools, LOLC maintains a shared clock and a shared library among musicians. Players synchronize by scheduling patterns to play at a specific beat and measure in the future. Whenever a pattern is defined, it is automatically shared with all other clients on the network. Once a pattern is defined, it is final and immutable in order to facilitate the creation of derivative patterns.

3.2.Pattern Creation

There are three ways to create a pattern in LOLC: a) single-element patterns can be defined using sound files; b) rhythmic repetitions of sounds are created through an event-definition syntax; and c) transformation operations modify and/or combine existing patterns.

Pre-recorded sound files serve as the base musical element in LOLC. Although any sound file can be used, short and percussive sounds tend to work best. Sounds are loaded as follows:

```
mySound : "sound.aif"
```

Another way to create a pattern is to specify the rhythmic repetition of an existing pattern through bracket syntax. For example, the following pattern definition plays mySound as two eighth notes at fortissimo, a quarter-note rest, one quarter note at mezzo-forte and four sixteenth notes at pianissimo:

```
myPattern : mySound[e.ff, e.ff, q.n, q, s.pp, s.pp, s.pp, s.pp]
```

If dynamics are omitted, mezzo-forte is assigned to the note. For a complete list of available durations and dynamics, see [8].

Patterns can also be nested. In the following example,

```
myPattern is played twice:
myNested : myPattern[w,h]
```

Each time the pattern is played, its note durations are stretched or compressed to match the target duration. In this example, the patterns remain unchanged for the whole note iteration and are halved for the half-note iteration.

Like Vocables [14], LOLC defines musical patterns as an ordered collection of items, but LOLC requires musicians to explicitly define rhythmic values, and it emphasizes rhythmic patterns that repeat single sounds. ixi lang [12] similarly emphasizes repetitions of single sounds and explicitly defined rhythms, but its grid-based approach to rhythm is both simpler and more constrained than LOLC's.

The last but most important way to create a new pattern is to transform or combine existing patterns. For all operations supported, see [8]. These operations were chosen because of their importance in studies on improvisational interaction [9] and musical pattern manipulations [19]. The syntax for all transformations follows this example:

```
pattern1 : sound1[w,h,h]
pattern2 : sound2[q,q,q,q]
myConcat:cat(pattern1, pattern2)
myTrunc : trunc(pattern2, 2)
```

The operation cat places two patterns in succession, so myConcat combines patterns based on two different audio-file sources. The operation trunc removes the final n items from a pattern, so myTrunc removes the final two quarter notes from pattern2.

3.3.Scheduling

Patterns are not played immediately upon creation. Instead, musicians must use an LOLC scheduling expression to determine when and how they play. Typically, patterns are scheduled for playback at the next beat or measure:

```
play myPattern @nextBeat
play myPattern @nextMeasure
```

LOLC supports additional scheduling methods: preview will preview the pattern over headphones and loop will play it repeatedly:

```
loop myPattern @nextMeasure ~16
```

This example loops the pattern sixteen times. Scheduling commands are shared with other clients via the text chat interface (Figure 1), but they are played solely on the local client machine.

3.4.Graphical User Interface

In LOLC's client software, commands are typed into an instant-messaging-style interface that shows both commands and chat messages from everyone in the ensemble (Figure 1). As musical patterns are created,

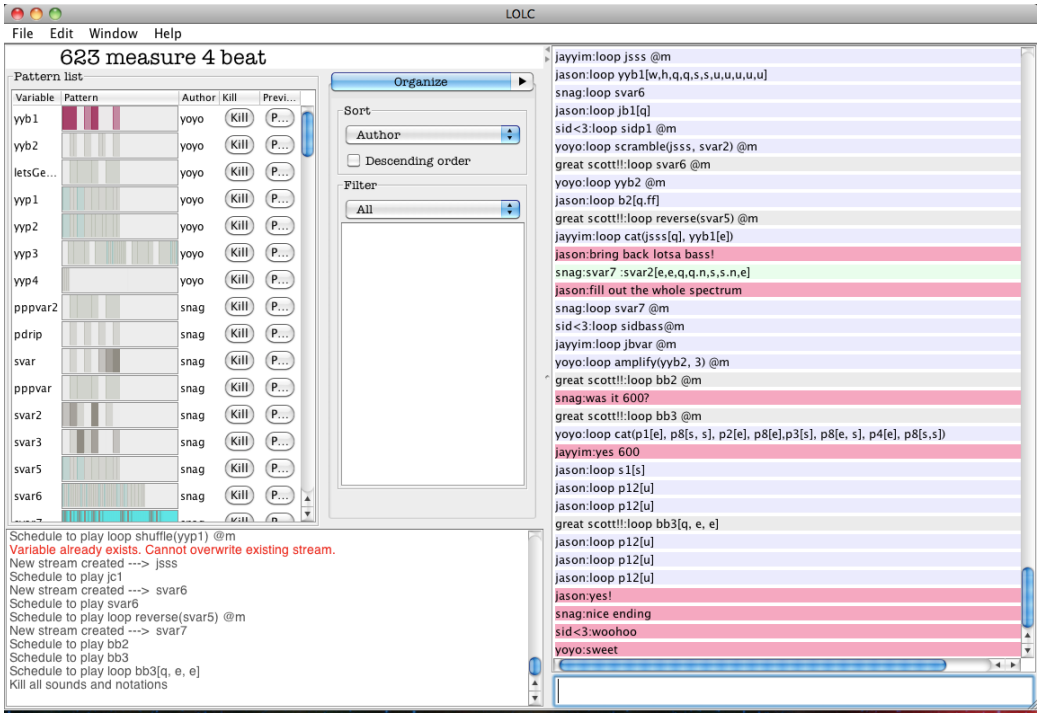

Figure 1 Screenshot of the LOLC client

they are automatically shared with the other musicians and displayed in a pattern library. Patterns are immutable once they are created so that they remain static as musicians build upon them, mimicking the collaboration of improvising acoustic musicians more than the dynamic, unpredictable shared objects of some laptop live coding environments. An info panel shows error messages in parsing and execution.

The pattern library panel on the left provides a visual interface for exploring ensemble activity, with the goal of facilitating closer collaboration and more pattern sharing among musicians. Patterns are displayed in the panel as a series of bars when they are created, along with a visualization of their content: the width of each bar corresponds to the duration of a sound, the height of a bar maps to the spectral centroid of a sound relative to the other sounds in the pattern, and the color of a bar is based on both the spectral and dynamic content of a sound. Variables are highlighted when they are played. Musicians can also sort and filter the list to isolate patterns that are created by certain musicians, created or scheduled at certain points in the performance, or based on particular audio file sources.

The LOLC server software includes a fullscreen visualization for projection to the audience. In the lower half of the screen, its visualization (Figure 2) shows all code and chat messages in a stylized, large-print format. In the upper half, each musician is visualized as a circle. The arcs between the circles represent the level of collaboration between the corresponding musicians. Chat messages are drawn in text balloons next to each circle. While a pattern is played, its rhythmic and spectral content is visualized as a series of concentric circles drawn outwards from the central circle.

4. EVALUATION AND DISCUSSION

To date, LOLC has been used in six musical performances. The musicians in most of these performances were graduate students in music technology, and some of them were also involved in the creation of LOLC. All were proficient programmers and computer musicians and many were experienced in live coding.

In April 2010, LOLC was used in a performance by the Princeton Laptop Orchestra (PLOrK); this performance is evaluated in detail in [8]. The undergraduate students who participated in this performance had more limited backgrounds in programming and computer music (and indeed, in music as well). But they still did not represent our ideal target group: highly-skilled musicians with little or no experience in programming.

This section, then, focuses on a single performance with LOLC in January 2011, in which musicians from a professional contemporary music ensemble performed with LOLC. These top-tier classical musicians have played with major symphony orchestras and also have considerable background with improvisation in experimental and/or jazz mediums. The laptop ensemble for this performance was

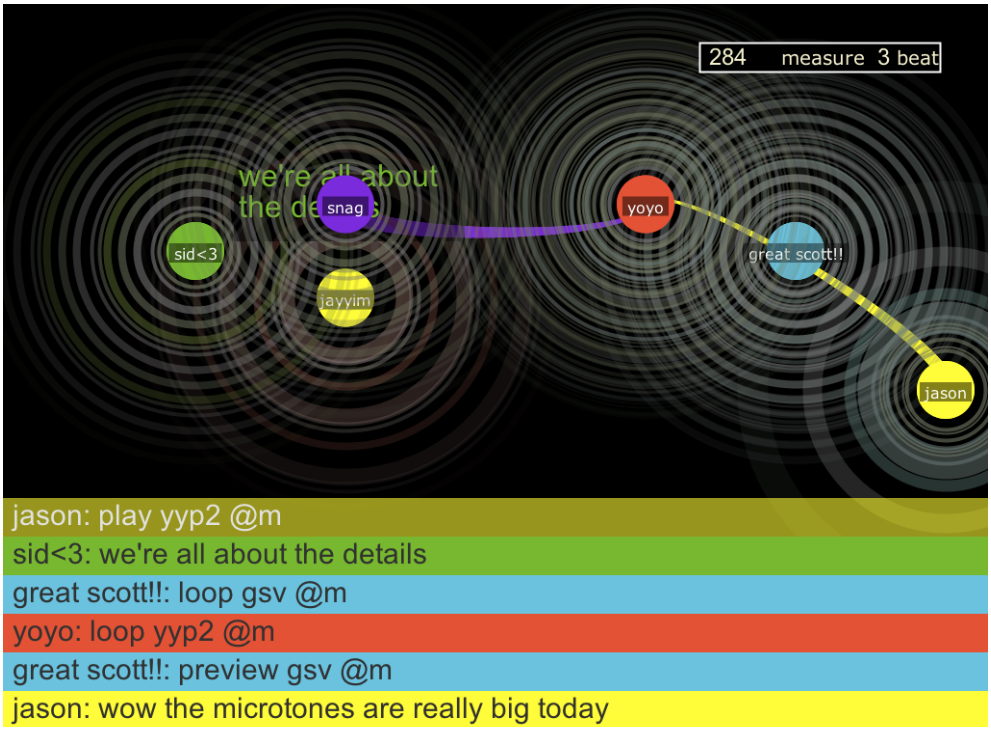

Figure 2 Screenshot of LOLC Server

composed of five members of the ensemble and one of the authors of this paper. Except for the author, the musicians had no prior experience with LOLC, no (or negligible) background in computer programming, and little or no experience with using computer music software.

One month before the show, the musicians started to learn LOLC through hour-long, one-on-one introductory sessions led by the authors. Each musician then practiced individually by following tutorial files and trying out LOLC on his or her personal computer. The ensemble rehearsed together for a total of 12 hours in preparation for the public performance. During the rehearsal process, we never instructed the ensemble about how to collaborate or how to structure the performance. We provided technical assistance and guidance on the environment, but let the ensemble develop their own structure for the improvisation and decide how to build the piece collaboratively.

Within this context, we used a variety of techniques to assess the degree to which LOLC succeeded in facilitating collaborative improvisation among the musicians. We logged code and chat messages to disk and analyzed each log quantitatively. Following the concert, we conducted an hour-long interview with each ensemble member to discuss his or her process of learning LOLC and experience of performing with it. The interviews particularly focused on how the musicians created music collaboratively. We also considered the musical output of the actual performance as well as of rehearsals.

Throughout the following evaluation, we exclude quantitative and qualitative data from the one member of the ensemble who is also an author of this paper and a developer of LOLC.

4.1.Collaborative Improvisation

Since LOLC's primary goal is to facilitate collaborative improvisation within a laptop ensemble, our evaluation focused on how successfully it did so. Since LOLC facilitates collaboration by encouraging musicians to borrow, re-use, and transform musical patterns from each other, one metric of collaboration is the degree to which patterns were shared among musicians.

In our analysis of the group's rehearsals and performance, we found a healthy level of collaboration among all musicians. During the dress rehearsal, for instance, the musicians created a total of 61 musical patterns; 19 of those patterns (31%) were based on patterns borrowed from others. The musicians scheduled a total of 117 different variables for playback; 37 of these patterns (32%) were borrowed from another musician. Although these statistics do not necessarily correlate to the quality of collaboration, they indicate a critical degree of sharing that is prerequisite to effective collaboration in LOLC, and they demonstrate that musicians were responding to each other through the music they improvised.

The amount and kind of sharing varied widely among the musicians. Figure 3 shows the number of patterns that were created and scheduled for each musician as a composite of borrowed (dark gray) and self-created ("owned") (light gray). Musicians A and E frequently borrowed patterns from other musicians (Figure 3a) but always transformed them before playing them; they never played a pattern directly borrowed from someone else (Figure 3b). In contrast, musician D never transformed a pattern created by another musician but often played others' patterns directly. Musicians B and C took a more balanced approach.

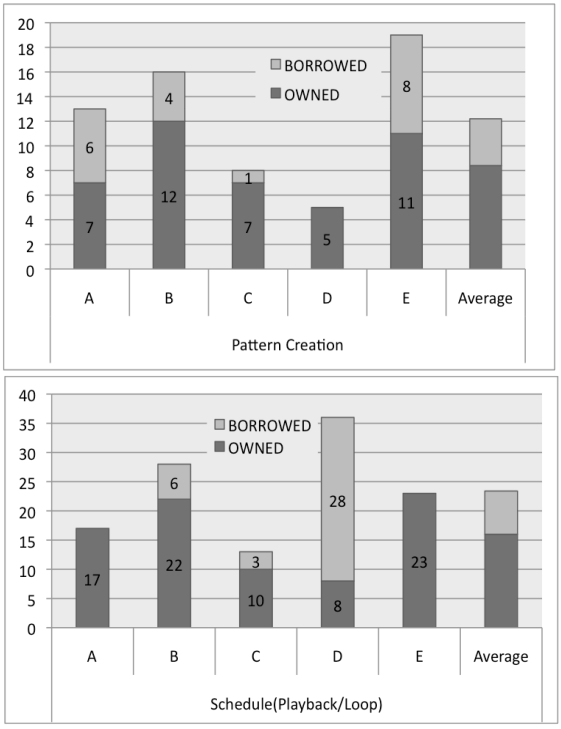

Figure 3 Number of patterns a) created and b) scheduled by each musician.

Considering this data in the context of musical improvisation, musicians A and E, who mainly created and transformed patterns, are those who introduce and develop musical ideas, while musician D listens to the patterns of others, picks them up, and responds to what others have played.

Even though different musicians have clearly adopted different roles within the ensemble improvisation, they did not determine or discuss these in advance. In interviews, we asked each musician

about their role within the ensemble. Each musician stated that there were no defined roles, though one musician noted that people did have their own signature "moves" (i.e. sound patterns) rather than roles. Also, most of the musicians noted their own particular techniques for transforming the material borrowed from others. But no one spoke of the ways in which each musician borrowed and transformed, which seems to have arisen more organically.

We also asked the musicians to describe the key ways in which they collaborated with the other musicians. The most common answer was to listen to what other musicians played and to transform or combine others' patterns. No musicians took advantage of all of the transformative operations available within LOLC; each relied on a handful of favorite operations (which varied from musician to musician).

Borrowing and transforming often became a chained process for the ensemble as musicians borrowed patterns that were themselves borrowed. One pattern from the dress rehearsal was based on eleven prior patterns (seven created from scratch) created by five different musicians in the ensemble over the course of nearly fifteen minutes. Figure 4 shows the chain of relationships among all of the variables as they were created, shared, and transformed over time. Table 1 shows the pattern definitions corresponding to each stage in the process. To us, LOLC's ability to facilitate the integration of ideas from so many musicians over such a long time scale attests to its potential to facilitate collaborative improvisation in an ensemble, using its ability to preserve the history of creation and collaboration within the performance to enhance the ability of musicians to borrow, transform, combine, and recall musical material.

Among the elements of LOLC, chat messages were the most effective in facilitating collaboration. One musician described how text chatting was used to structure the piece and to keep everyone thinking together (as opposed to each musician working in isolation). The group's only advance planning as to structure was the broad idea of a sparse beginning and gradual buildup to a sudden ending. All other details, such as timbre, density, and pacing, were coordinated on the fly through the chat window as musicians posted messages such as "sounds like we are winding down, let's go to all bass" and "ease off a bit or keep goin?" One interviewee also noted that the text chatting helped him focus on listening more and to ease the confusion coming from the unfamiliarity of his new instrument, the laptop.

The projection shown to the audience, which visualized the collaboration among musicians as lines connecting them together, also encouraged and supported improvisation. During the interviews, two musicians mentioned the visualization as a way of helping them determine with whom they had not yet interacted and from whom they wished to borrow in the future.

The musicians were proud of their ability to collaborate together musically in performance, and about the progress they made in this regard over the course of their rehearsals. In an interview, one said: "I remember the first rehearsal and I certainly was creating in a vacuum and the five other people were so as well. People were throwing out interesting things that they had come up with. As it went on, I was happy to get to the point where I was actually understanding, was able to listen, figure out what's going on and was able to do with my ears as well as eyes."

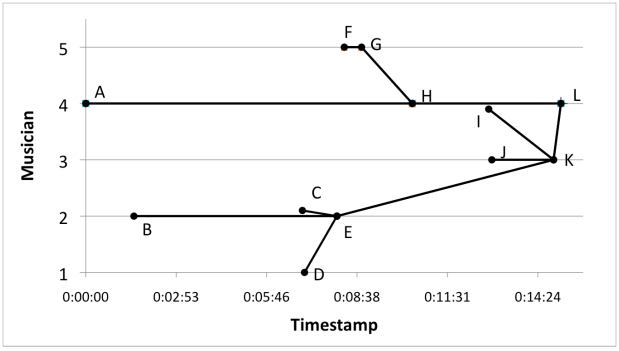

Figure 4 The chain in creating the pattern "tls7"

	Expression
A	jam : cat(b3[e,e,h.n,q.n,e,e,q.n],p43[e.n,e,s.n,s,e.n])
B	sangre:b4[e,e,e,e.n,q,q,s,s,s.n]
C	sangre1:h1[u,u,u,h.n,e,e,h.n]
D	snark: s1[e.fff,e.fff,s,s,w.n]
E	sangre2:cat(snark, sangre1, sangre)
F	hades:cat(b2[s.p,s.p,s.mp,s.mp,s.mf,s.mf,s.f,s.f],b5[h.ff,q.n])
G	newhades:mirror(hades)
H	jam6: cat(jam,newhades)
I	tls1 : cat(b1[e],p40[s,s],b1[e])
J	stutter2 : s2[u,u,u,u,u,u,u,u, w.n, h.n]
K	big1 : shuffle(cat(sangre, sangre2, tls1, stutter2))
L	tls7 : cat(big1,jam6)

Table 1 Pattern creation expression at each point in Figure 4.

4.2.Comparison to Instrumental Improvisation

Since the musicians in this performance all had extensive experience improvising on traditional acoustic instruments in jazz and/or avant-garde settings, we wanted to understand how the experience of improvising with LOLC on a laptop differed from

the more traditional modes of improvisation that inspired LOLC.

In the interviews, the biggest difference noted by the musicians was the inherent latency of interaction. One musician described the difference as follows:

> *I would always have a minute delay from the time I figure out what I want to do until I actually could get the computer to play what I wanted to do.*

Part of this stems from typing, but there is a more fundamental difference here in how sounds are generated in each context. One musician noted that with a conventional instrument, playing is second nature and becomes a natural extension of musical ideas, while in LOLC it takes time to write a script to play what is intended. There is also a step of translating musical ideas into a symbolic, notated form, as one musician noted:

> *I got pretty good at what the rhythm is, for example, e stands for eighth notes, and how to translate that language notation-wise.*

Not only must a player translate the music in his mind to the corresponding syntax, but he also must think about the music in terms of traditional parameters of musical notation that correspond to its symbolic representation in LOLC. In contrast, improvisation with traditional instruments tends to require less notation-based thinking and more listening to sounds and responding immediately with motor memory [17]. We expect some of the sub-tasks, like making rhythmic patterns, could become more automatic with enough practice, but still will not approach the level of perceptual motor skills [16].

The inherent delay in textual environments like LOLC can be a serious drawback to facilitating responsive collaborative improvisation, especially in the conversational style of jazz improvisation. However, other aspects of LOLC differ from instrumental improvisation and offer unique advantages. One member pointed out that LOLC is fundamentally different because in LOLC you schedule something as many times as you want, let it go and you are free to do something else while it is being played. In other words, LOLC permits a single player to instantiate many simultaneous, independent layers because each can continue automatically once it is scheduled. In contrast, when a musician improvises with a conventional instrument, he or she must constantly be engaged in the production of each individual event. LOLC, like many laptop music environments, enables musicians to automate the management of these low-level musical events and focus on higher-level decisions [6].

Another musician pointed out that in LOLC, you could literally pick or steal fragments from other people and reproduce them identically and promptly. In contrast, in the instrumental context, a player has to listen to the material and try to reproduce it.

The musicians also commented on the nature of the transformation operations supported by LOLC. Many of these operations involve a degree of randomness. One musician had trouble imagining how the transformed pattern would sound; this was ultimately not a problem but a benefit: it led the musician to improvise with a novel mindset. Instrumental musicians often believe that unpredictability comes from a lack of musical or technical proficiency. But in LOLC, they transition into an environment where the one-to-one mapping of a traditional instrument is sometimes replaced by a one-to-many mapping in which a gesture can trigger a complex set of events, much as in many computer music interfaces [10]. The musicians in this performance made this transition successfully, adapting to one-to-many and sometimes non-deterministic ways of creating material.

We have previously noted that musicians tend to use LOLC to create loop-based music with slowly evolving musical textures [8]. It appears that the inherent differences of LOLC from traditional instrumental improvisation contexts push musicians to play this particular style of music.

On the other hand, some interviewees pointed out that both types of improvisation have much in common. One musician said:

> *To me it was a similar feeling of what it feels like if I was performing a piece of music on a normal instrument. I am trying to make something special in the moment for that performance. To me, it still has the same basic musicianship issue, trying to be listening to others, to be responsive to what's going on around you, and not just to focus on yourself. To me it was more of trying to take your normal experience of making music and applied it to this program rather than having the program somehow change the way I make music.*

All musicians stated that they enjoyed the improvisation with LOLC as they do with conventional instruments.

5. CONCLUSION AND FUTURE WORKS

In conclusion, LOLC has largely succeeded in facilitating collaborative improvisation among a target group of musicians with little background in computer music and programming. In addition, although the musicians found that LOLC is fundamentally different from improvisation with conventional instruments, they made successful transitions.

This year, we have also focused on a new area of exploration with the LOLC environment. We extended LOLC to the realm of real-time notation, in which musicians sight-read conventional or graphical notation live, in performance, as it is rendered on a

digital display [7]. In this scenario, LOLC musicians can manipulate musical score fragments in addition to audio files, and those fragments are displayed in real time to sight-reading musicians (on traditional instruments) with whom each laptop musician is paired. Given the unique benefits of text-based laptop improvisation with LOLC and performance on traditional instruments, we find the integration of both modes within a single performance environment to be particularly exciting. In February 2012 we presented an initial performance using LOLC for real-time notation, and a comprehensive study of the system is currently underway.

6. ACKNOWLEDGEMENTS

LOLC is supported by a grant from the National Science Foundation as part of a larger research project on musical improvisation in performance and education (NSF CreativeIT 0855758). We also thank all the musicians of Sonic Generator. Additional information is available at http://www.jasonfreeman .net/lolc/

7. REFERENCES

[1] Allen, S. Teaching Large Ensemble Music Improvisation. *Radical Pedagogy* 4(1). Available at radicalpedagogy.icaap.org/content/issue4_1/ 01_Allen.html. 2002.

[2] Berliner, P. Thinking in Jazz: The Infinite Art of Improvisation. Chicago: *University Of Chicago Press*. 1994.

[3] Brown, A., and A. Sorensen. Interacting with Generative Music through Live Coding. *Contemporary Music Review 28*,1(2009), 17–29.

[4] Brown, C., and J. Bischoff. Indigenous to the Net: Early Network Music Bands in the San Francisco Bay Area. Available at crossfade.walkerart.org/ brownbischoff, 2002.

[5] Collins, N., et al. Live Coding in Laptop Performance. *Organised Sound 8*,3(2003), 321–330.

[6] Collins, N. Generative Music and Laptop Performance. *Contemporary Music Review* 22, 4 (2003), 67-79.

[7] Freeman, J. Extreme Sight-Reading, Mediated Expression, and Audience Participation: Real-time Music Notation in Live Performance. *Computer Music Journal 32*, 3 (2008), 25–41.

[8] Freeman, J. and Troyer A.V. Collaborative Textual Improvisation in a Laptop Ensemble. Computer Music Journal 35, 2(2011), 8-21.

[9] Hodson, R. Interaction, Improvisation, and Interplay in Jazz. Routledge, New York, 2007.

[10] Keislar, D. A Historical View of Computer Music Technology. In R. Dean (ed.), *The Oxford Handbook of Computer Music*. New York: Oxford University Press, 2009, 11–43.

[11] Lewis, G. *Virtual Concerto*. Unpublished musical score. 2004.

[12] Magnusson, T. The ixiQuarks: Merging Code and GUI in One Creative Space. *In Proceedings of the International Computer Music Conference* 2 (2007), 332–339.

[13] McCartney, J. Rethinking the Computer Music Language: SuperCollider. *Computer Music Journal* 26, 4 (2002), 61–68.

[14] McLean, A., and G. Wiggins. Words, Movement and Timbre. In *Proceedings of the 9th International Conference of New Interfaces for Musical Expression,* 2009.

[15] Monson, I. Saying Something: Jazz Improvisation and Interaction. Chicago: *University of Chicago Press*. 1996.

[16] Nilson, C. Live Coding Practice. In *Proceedings of the 7th International Conference on New Interfaces for Musical Expression*, 2007.112–117.

[17] Pressing, J. Improvisation methods and models. In Sloboda, J.A. (ed.), *Generative Processes in Music: Psychology of Performance Improvisation and Composition*. Oxford: Clarendon Press. 1988. 129–178.

[18] Rohrhuber, J. Networked Programming. Available on-line at supercollider.svn.sourcefo- rge.net/viewvc/supercollider/trunk/common/build/ Help/Libraries/JITLib/tutorials/jitlib_networking.h tml?revision=10439.

[19] Spiegel, L. Manipulations of Musical Patterns. *In Proceedings of the Symposium on Small Computers Q7 and the Arts*, 1981. 19–22.

[20] Wang, G., and P. Cook. ChucK: a concurrent, on- the- fly audio programming language. In *Proceedings of the International Computer Music Conference,* 2003. 219–226.

[21] Wang, G., et al. CoAudicle: A Collaborative Audio Programming Space. In *Proceedings of the International Computer Music Conference*, 2003. 331– 334.

[22] Zorn, J. John Zorn's Cobra: Live at the Knitting Factory. New York: Knitting Factory Works KFW124, compact disc. 1995