LUSH: An Organic Eco+Music System

Hongchan Choi
Musical Art and Technology Laboratory (MARTE)
Dongguk University
Jung-gu, Pil-dong 3-26, Seoul, Korea
hongchan@dongguk.edu

Ge Wang
Center for Computer Research in Music and Acoustics (CCRMA)
Stanford University
660 Lomita Dr. Stanford, CA 94305
ge@ccrma.stanford.edu

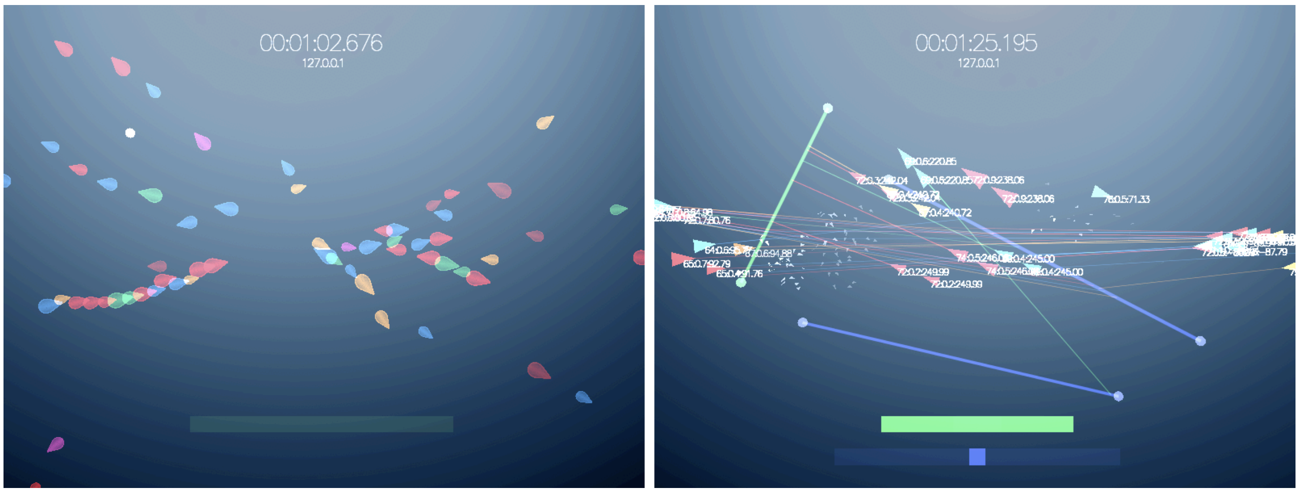

Figure 1. The sea of Lush. A colorful blob called boid is considered to be identical to a musical note. (left) Movable playheads can be exists simultaneously to play various musical boids with sonic properties. (right)

ABSTRACT

We propose an environment that allows users to create music by leveraging playful visualization and organic interaction. Our attempt to improve ideas drawn from traditional sequencer paradigm has been made in terms of extemporizing music and associating with visualization in real-time. In order to offer different user experience and musical possibility, this system incorporates many techniques, including; flocking simulation, nondeterministic finite automata (NFA), score file analysis, vector calculation, OpenGL animation, and networking. We transform a sequencer into an audiovisual platform for composition and performance, which is furnished with artistry and ease of use. Thus we believe that it is suitable for not only artists such as algorithmic composers or audiovisual performers, but also anyone who wants to play music and imagery in a different way.

Keywords

Algorithmic composition, behavior simulation, automata, music visualization, music sequencer, musical interface, audiovisual frameworks.

1. MOTIVATION

The foremost motivation of this system is to defy the stereotypes of conventional music sequencers and suggests an alternative that provides more “organic” interaction and combines visualization and probabilistic elements aesthetically. We achieved the investigation by renovating the visual facet of the sequencer and adding features such as algorithmic music generation and networking.

Since the music sequencer was invented around 1950, it has played a central role in music production. From the analog sequencer equipped with dozens of relay circuits, switches and knobs to the software sequencer for Atari personal computers, musicians and engineers have constantly pursued to enhance and complement its shape and feature. The advent of computer-based sequencers at early 1980s had largely contributed to the productivity and efficiency in music creation by assisting musicians to manipulate musical events in a visual way. Thanks to the introduction of graphic user interface (GUI) and ever-increasing computing power, the capability of the traditional sequencer was expanded by integration of audio editing, recording and even sound synthesis.

Although numerous variations of music sequencers have emerged throughout the computer music history, most of them have two elements in common; a system-wide playhead and the visual representation of discrete notes to be played. They are useful and informative to the user in view of providing the flexibility of non-linear editing. These visual cues improve the overall workflow in general, however, they become crucial restraints particularly when creating a generative piece or improvisation. We believe music creation process continues to

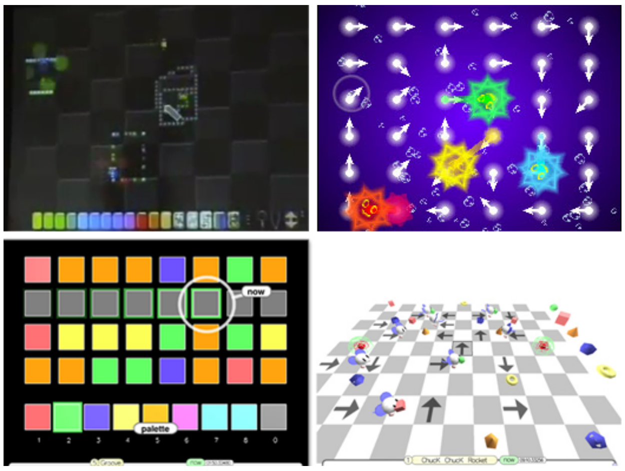

Figure 2. Music Insects and Electro Plankton (above) and Graphical Interfaces in Audicle (below)

change in order to be more spontaneous as well as interactive; this idea will shift the traditional paradigm of the music sequencer ultimately.

Lush, as Eco+Music system we claim, embodies the following ideas:

- Highly visualized environment tailored for interaction and improvisation, proposing new musical entities of movable playheads and notes like living organism.
- Exploration of countless possibilities by generating music sequence from nondeterministic finite automata (NFA), built by analyzing a score data. (e.g. a standard MIDI file) This also offers chances to reuse and visualize a vast collection of score files.
- Supporting OSC(OpenSound Control) allows the system to function as visual front-end or control interface for OSC-compatible music software as well as to collaborate with other Lush system by communicating generated data over the network [10].

In sum, we incorporate several features such as interaction, visualization, algorithmic composition and networking into one system bearing aesthetic and usability in mind.

2. EXISTING RELATED WORKS

A variety of audiovisual systems that influenced and inspired us are derived mainly from three areas: hybrid music sequencer, such as Music Insects and Electro Plankton (figure 2) [1] created by Toshio Iwai, graphical/collaborative music interfaces such as Audicle (figure 2) [2], and algorithmic music generation.

Iwai Toshio's Music Insects and Electro Plankton, the most influential examples to our system, realize a series of novel concepts in software by transforming the visual limitation that traditional music sequencers have.

Various graphical/collaborative interfaces in Audicle also inspired our work, especially in terms of music generation and network collaboration. The ideas of "electronic chamber music", pioneered by the Princeton Laptop Orchestra (PLOrk) and Stanford Laptop Orchestra (SLOrk), introduced a new form of musical environment based on innovative visual representation and collaboration. In addition, Rob Hamilton's Q3OSC is also similar with our system in several aspects; it is derived from 3D gaming platform supports animated visual, dynamic interaction and network gaming [3].

Making music with stochastic algorithm, which is the essential part of our system, has a long-standing history. It is partly due to the accessibility of stochastic algorithms that they have become so extensively employed for compositional purpose. The CAMUS 3D system is an example of musical application leveraging stochastic quality of cellular automata [8] and it can be labeled as the first standalone software for algorithmic composition.

Golan Levin has pioneered the multi-modality for audiovisual system in his remarkable projects. Levin's Painterly Interfaces [4] have introduced a new sense of audiovisual aesthetic focused on organic and natural properties. Scrapple, the one of his sound installation, also suggests how the music sequencer can be evolved in collaborative and tangible way [5]. His insight to simplicity and organic senses has largely motivated and informed our exploration.

Additionally, tangible music sequencers such as Tenorion created by Toshio [6] and Siftable created by David Merrill [7] also introduced the new way of music making. These works are not related to Lush directly, however, their strong implication to new approaches for music making can be considered our next step.

3. USER EXPERIENCE

The user can use this system for either casual or creative purpose. First of all, Lush is designed to provide multi-colored animated visualization of the source score file so that one can simply enjoy its motion graphics. Secondly, the user can control playheads using mouse and keyboard to hit boids, which are graphical objects equivalent to discrete musical notes. This interaction results music creation as well as sonification of musical data from the original score file. In the end, the user can send control data generated by interaction to other OSC-compatible applications such as ChucK, PureData, and Max/MSP [11][12][13].

We deployed several visual entities into our work to realize "musical eco system", including; playhead, playqueue, flock, and boid. We believe this approach makes Lush more distinguishable than traditional sequencers in terms of user experience. The following section details what they are and how to use them.

3.1 Playheads and Playqueues

The playhead in Lush is the most significant difference from the conventional sequencer. There is no geometric constraint for the playhead; it can be located, rotated, resized and even duplicated on the screen without any restriction. When the user starts dragging the mouse, the one end of the playhead will be anchored and the other end will be moving along one's control. This pivotal gesture enables many different ways to hit boids. The user can select the one of multiple playheads by pushing number keys 1 through 5. The playqueue collects collision events in the system and play them with the integrated synthesizer or send them over the network via OSC connection. They are animating whenever their contents change to let the user know how many notes are playing.

Lush can have up to 5 playheads and 2 playqueues simultaneously (figure 4). One green playhead/playqueue are instantiated by default in order to play the built-in synthesizer. In contrast, the rest of playheads and playqueue are designed to send data of played notes to local or remote applications. For a

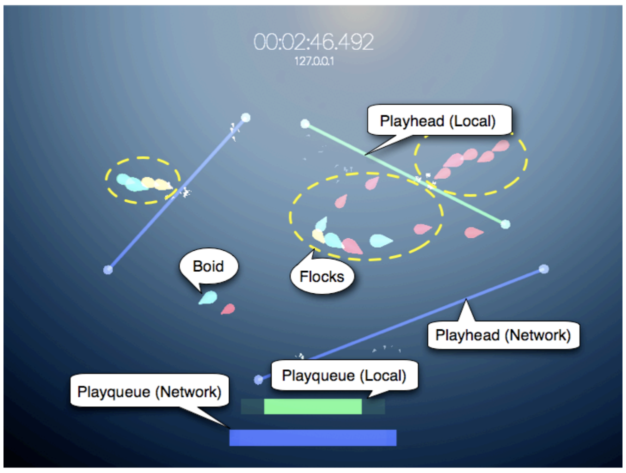

Figure 4. Two types of playhead and playqueues. The green ones are to play sound with the built-in synthesizer, and the blue ones are to send control data to OSC-compatible software. (e.g. ChucK, PureData, Max/MSP, and etc.)

boid to be played it must be collided with any playhead first and then it is queued in the appropriate playqueue.

3.2 Flocks and Boids

A "boid" is a minimal unit in the system, which is equivalent to an individual note in traditional sequencers. A boid can have many different audiovisual attributes including pitch, amplitude, color, position, size, and velocity vector (figure 4). They can be played as musical notes when colliding with any playhead. A "flock" is a group of boids and we see it as a musical phrase. (flocks in figure 5 are circled with dashed lines.)

The intriguing fact about boids and flocks is their movement is not totally random, but very organic and natural, thank to a special behavior simulation algorithm called "flocking algorithm." Flocking is the behavior exhibited when a group of birds, called a flock, are foraging or in flight. Craig Reynolds first simulated this behavior in 1986. This algorithm simulates simple agents (boids) that are allowed to move according to a set of basic rules. The result is akin to a flock of birds or a school of fish [9].

4. THE LUSH ARCHITECTURE

The Lush architecture is incorporating various technologies into one system based on C++ programming language. We have implemented Lush as a standalone application, while designing it readily available to any type of projects necessitate real-time visualization based on the particle system. We believe the particle system is the most powerful way to create sound visualization because of its parametric trait and adjustable degree of randomness.

4.1 C++ Class Hierarchy

In terms of its architecture, Lush is hierarchically abstracted and structured. The aforementioned boid is the smallest object in the system and its instances calculate own velocity vector and collision at every frame, in other words, it is an autonomous entity. Flocking is considered an emergent behavior arising from simple rules that are followed by individuals and does not involve any central coordination.

The "flock" class is a container class for boid instances of which main tasks are iterating processes of boids. This mechanism provides flexibility and performance to the system. Grouping multiple boids by flocks offers two advantages. First of all, the user recognizes it as a musical phase due to its visual representation. Secondly, flocking simulation normally requires huge amount of calculation as it takes account of every boids in the system, however, it is possible to reduce computational load by making them interact only with ones in the same flock.

Likewise, the "sea" class is a container as well as an iterator of flock instances. It is also the biggest object (singleton class) in the system that embraces everything inside including playheads and playqueues (figure 5).

4.2 Vector Calculation and Flocking

In the Lush, the collision is the most important event in either musical or visual aspect. A custom class for convenient vector calculation has been implemented to solve collision and intersection. It plays a significant role in operating flocking algorithm as resulting a number of parameters to be tweaked. Besides, all of these parameters are readily available to visualization. The "vector-madness mode" is a good example of visualization utilizing one of these parameters – a steering vector of a boid, which is the most interesting parameter to watch in the system (figure 1 right).

4.3 Nondeterministic Finite Automata

Lush generates music from nondeterministic finite automata, and this is the most effective and easiest way to create musical sequence algorithmically while maintaining melodic/scale contour of the original music. The system analyzes a standard MIDI file, only for the pitch of notes, to build automata and assign weighting factors to links between states by the number of appearance of MIDI notes.

4.4 Integration: OpenGL + STK + OSC

OpenGL is the core of the system that drives other parts. STK (Synthesis TookKit) is deployed for MIDI feature and the built-in synthesizer. In addition, RtAudio functions as the audio API allowing the system to communicate with the audio interface in

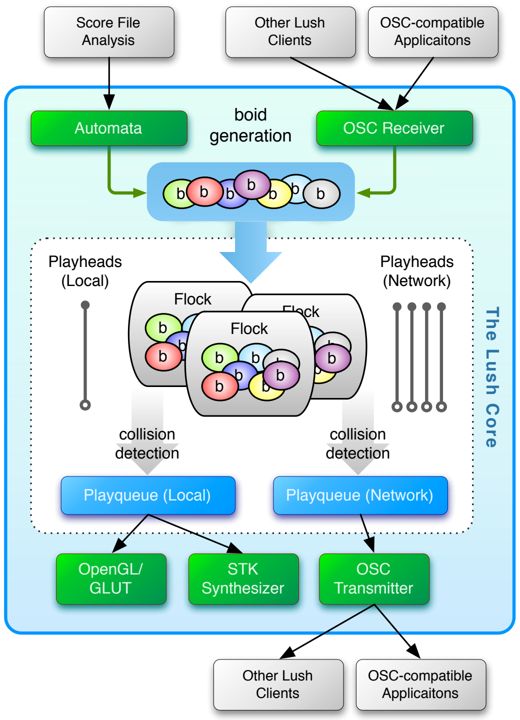

Figure 5. The Lush Architecture.

the computer [14] [15]. Ross Bencina's OSCPack is also used to implement OSC communication [16].

5. ANALYSIS AND FUTURE WORK

We believe this system offers algorithmic composers countless possibilities to interact with generated music sequence in real-time. Also its visual aesthetic motivates audiovisual performers to present their working process to audiences as a modern form of abstract audiovisual art. Furthermore, the intuitive user interface of Lush has the potential to evolve into music games or audiovisual games for the non-musician population.

5.1 Limitations

A number of disadvantages have been found in actual practice. First of all, it is impossible to play notes along the musical time base. (e.g. measures, beats and ticks) The other one comes from its incomplete analysis of score files. Currently, NFA cannot convey the complete characteristics of original music because of the analysis process bypassing some of the crucial parameters such as duration, velocity and pause in score data. Lastly is the lack of a recording feature. We found that many users wanted to record or capture their audiovisual performance or composition while using the system.

5.2 Contributions

However, Lush has a number of novelties to improve limitations that are commonly found on traditional sequencers. Compared to their disinterest on visual aesthetics, Lush provides a greater level of visualization and responsive interaction. On account of NFA, the user can balance the characteristics of its musical product between stochastic and deterministic quality by selecting notes from computer-generated music sequence. In addition, the practical observation of the system shows playing boids in flocks can be considered as sonification of flocking algorithm. The color distribution of generated music sequence is one of the interesting points. Finally, based on these advantages, it offers chances to reuse a huge collection of MIDI files while generating music sequences with a similar sense of the original music.

5.3 Future Work

We realized building NFA by analyzing existing score files has the potential in it, so enhancing and complementing it is the top priority of our next step. The secondary task is to make the system communicate with the other one over the OSC connection. (e.g. ChucK-ChucK Rocket [17], SbLAC [18]) Furthermore, more components provoke user's interaction and inspiration will be added. It includes more intelligent behavior patterns for boids, force field, warp hole, and etc.

6. CONCLUSION

We developed a music creation environment by incorporating interaction, visualization, and aesthetic. This system offers various functionalities that conventional sequencers lack including music sequence generation based on score file analysis, visualization of musical notes by behavior simulation and networking for the other OSC-compatible applications. We propose Lush as a playground for algorithmic composers and audiovisual performers, promoting this environment as a starting point for the concept of "performative composition" or "compositional performance."

7. PROJECT URL

http://ccrma.stanford.edu/~hongchan/lush/

8. ACKNOWLEDGMENTS

We appreciate the valuable feedback from the people in MUSIC 256a class (Music, Computing and Design I) at CCRMA.

9. REFERENCES

[1] Iwai, T. "Images, Music, and Interactivity - the Trace of Media Art" *Keynote Speech. International Computer Music Conference*. June 2004.

[2] Wang, G. "The Audicle: A Context-Sensitive, On-the-fly Audio Programming Environ/mentality" *In Proceedings of the International Computer Music Conference*. June 2004.

[3] Hamilton, R. "Q3OSC Or: How I Learned to Stop Worrying and Love the Game" *In Proceedings of the International Computer Music Conference.* 2008.

[4] Levin, G. *Painterly Interfaces for Audiovisual Performance.* M.S. Thesis, MIT Media Laboratory, August 2000.

[5] Levin, G. "The Table is The Score: An Augmented-Reality Interface for Real-Time, Tangible, Spectrographic Performance" *In Proceedings of the International Computer Music Conference.* 2006.

[6] Yamaha Corp. "Tenori-on" http://www.global.yamaha.com/design/tenori-on/ Retrieved January 2010.

[7] Merrill, D., Kalanithi, J., Maes, P. "Siftables: towards sensor network user interfaces" *In Proceedings of the 1st international conference on Tangible and embedded interaction.* 2007.

[8] McAlpine, K., Miranda, E., Hoggar, S. "Making music with algorithms: A case-study system" *Computer Music Journal. Vol. 23, No. 2.*1999.

[9] Reynolds, C. W. "Flocks, Herds, and Schools: A Distributed Behavioral Model" *In Computer Graphics (Proceedings of SIGGRAPH '87).* 1987.

[10] Wright, M. and Freed, A. and Momeni, A. "Opensound control: State of the art 2003" *In Proceedings of the conference on New interfaces for musical expression.* 2003.

[11] Wang, G. *The ChucK Programming Language: a Strongly-timed, On-the-fly Environ/mentality*. PhD Thesis. Princeton University Press. 2008.

[12] Puckette, M. "Pure Data: another integrated computer music environment" *In Proceedings of the Second Intercollege Computer Music Concerts.* 1996.

[13] Cycling 74'. http://www.cycling74.com Retrieved January 2010.

[14] Cook, P.R. and Scavone, G. "The synthesis toolkit (stk)" *In Proceedings of the International Computer Music Conference.* 1999.

[15] Scavone, G.P. and Cook, P.R. "RtMidi, RtAudio, and a synthesis toolkit (STK) update" *Synthesis Journal*. 2004.

[16] Bencina, R. http://www.audiomulch.com/~rossb/code/oscpack/ Retrieved January 2010.

[17] G. Wang and M, Ananya and Cook, P.R. "Building Collaborative Graphical interFaces in the Audicle" *In Proceedings of the International Conference on New Interfaces for Musical Expression.* 2006.

[18] Choi, H. http://ccrma.stanford.edu/~hongchan/sblac/ Retrieved January 2010