FARM 2016 Concert

Live Music and Visuals Produced through Functional Programming

Renick Bell
independent (Japan)
renick@gmail.com

Abstract

A concert of performances employing functional programming techniques will take place at FARM 2016 in Nara, Japan. Eight performances will be presented on a large full-range sound system with video projection on two walls. Performances will include live coding and generative systems used for both audio and visuals.

Categories and Subject Descriptors J.5 [*Performing arts*]

Keywords functional programming, music, art, visuals, live coding, generative art, Haskell, Scheme, TidalCycles, SuperCollider

1. Concert Overiew

A concert of performances employing functional programming techniques will take place at FARM 2016 in Nara, Japan. While performances have taken place as part of previous FARM workshops, this was the first year that a separate call for performances was made. The call sought "proposals for live performances (audio, visual, or both) which employ functional programming techniques in whole or in part" and encouraged "both risk-taking proposals which push forward the state of the art and refined presentations of highly-developed practice".

The concert takes place from 19:30 to 21:30 at a live house in Nara called Beverly Hills. The sound will be diffused through a fairly high-volume full-range PA system. The audio will be generated mostly through functional programming techniques. Those techniques will be used in live coding and are present in the generative systems to be presented (with the exception of Yullippe's ambient/techno live performance accompanying the performance of a system for generated visuals). The musical genres represented include art music, noise, experimental music, ambient, techno, and chiptunes. There will be two projectors on which various visuals will be presented, ranging from code and an interpreter in the case of live coding to system visualization to rendered graphics.

Performances will be presented by the following performers, listed alphabetically by surname or group name:

- Renick Bell
- Alexandra Cardenas
- Atsuro Hoshino
- Akihiro Kubota
- Jay McCarthy
- RAW (a duo of Seluk Artut and Alp Tuan)
- Francis Gene Shuman with Yullippe
- Atsushi Tadokoro

There are some commonalities between performers regarding systems and languages used for performances.

- TidalCycles [10] (used by Cardenas, Kubota, Tadokoro)
- SuperCollider [9] (used by Bell, Cardenas, Hoshino, Kubota, RAW, Tadokoro)
- Haskell [12] (used by Bell, Cardenas, Kubota, Tadokoro)
- Scheme [5] (used by Hoshino, McCarthy)

Haskell and Scheme require no introduction. In addition to those, Shuman uses PureScript, a Haskell-inspired language. [6] Some may be unfamiliar with the music systems above. TidalCycles (formerly just Tidal) is a domain-specific language which runs in the Glasgow Haskell Compiler interpreter.[7] It is used to generate patterns of values which can represent samples, rhythms, and other performance parameters. It then outputs those patterns to a synthesizer. SuperCollider is a synthesis system consisting of a synthesizer called scsynth and a multi-paradigm programming language specifically developed for scsynth called sclang. Five out of six of the performances employing SuperCollider do so through clients rather than using sclang directly or bypassing sclang.

What follows is a list of performers with brief notes edited from performer submissions about their performances and bios.

2. Performance Notes and Bios

2.1 Renick Bell

Renick Bell will do a live-coded performance with his own library, called Conductive, for instantiating agent processes and generating patterns which those agents follow. [2] By manipulating those agents, which trigger a sampler or control other agents, a rapidly changing stream of bass, percussion, noise, and tones is improvised according to a rough sketch of the overall performance structure. The sample player was built with hsc3, a Haskell client for SuperCollider by Rohan Drape. [3] Interaction with the system, which is projected for the audience, employs the Glasgow Haskell Compiler Interpreter (ghci), the vim text editor, the xmonad window manager, and the tmux terminal multiplexer.

Bell is a computer musician, programmer, and teacher living in Tokyo, Japan. He is a graduate of the doctoral program at Tama Art University in Tokyo, Japan. His current research interests are live coding, improvisation, and algorithmic composition using open source software.

2.2 Alexandra Cardenas

Alexandra Crdenas will perform through live coding, combining her interests in improvisation, composition, programming, live electronics and traditional music. Alexandra projects her screen for the audience to witness what she is writing on her computer. Using SuperDirt (a SuperCollider implementation of the Dirt sampler for the TidalCycles programming language) Alexandra creates her own sounds in SuperCollider and sequences them using patterns written in real time with the software TidalCycles.

Composer, programmer and improviser of music, Crdenas has followed a path from Western classical composition to improvisation and live electronics. Using open source software, her work is focused on the exploration of the musicality of code and the algorithmic behaviour of music, especially through live coding. Currently she lives in Berlin, Germany and is doing her masters in Sound Studies at the Berlin University of the Arts.

2.3 Atsuro Hoshino

Atsuro Hoshino will live code audio using GNU Emacs for textual user interface, SuperCollider for audio synthesis engine, and Scheme code executed in GNU Guile for gluing things together. The Scheme code will use the rsc3 library to interact with SuperCollider. [4] Sequential events are controlled with a technique called temporal recursion. [11] Like any other recursive function, temporal recursion is a recursive function defined in Scheme, but taking a time stamp as an argument. This form enables asynchronous updates of the body of a function without interrupting audio events sent to synthesis engine.

Hoshino is a software engineer in Tokyo, Japan who since university has been playing with various computer music languages. After working with common development languages at a startup company, he encountered Haskell and fell in love with it, though his recent interest is growing toward the LISP language family.

2.4 Akihiro Kubota

Akihiro Kubota will perform a new kind of sound poetry using sound data from the world's first art satellite, ARTSAT1:INVADER (http://artsat.jp). INVADER was equipped with Morikawa, an onboard mission computer compatible with the Arduino open-source hardware platform. Morikawa's missions included algorithmic generation and transmission of synthesized voice, music and poems, capturing and transmitting of image data and communicating with the ground through a chatbot program. The fragmented sound data is reconstructed as a live coding performance using TidalCycles. Kubota finds the flexible and multiple notations of this functional language to be very useful for real-time (live coding) performance.

Kubota is a professor of the Art and Media Course in the Information Design Department at Tama Art University in Tokyo, Japan. He earned his doctorate at the University of Tokyo in the Faculty of Engineering. His projects, such as the ARTSAT Project (the world's first nano art satellite and 3D-printed artwork to be successfully launched into deep space) incorporate the diverse fields of bio art, digital fabrication, sound performance and the creation of original musical instruments.

2.5 Jay McCarthy

Jay McCarthy will present a performance of Bithoven, a composer of approximately 1.079363e+239 different compositions based on four-part harmony and basic chord progressions. It is combined with a purely functional audio synthesis engine based on the Ricoh RP2A03, found in the 1985 Nintendo Entertainment System (NES). The synthesis engine is parameterized over a band of instruments and styles of play, so that each composition can be played in one of approximately 4.22234e+41 different arrangements or "NEStrations". The music is thought to be plausible to most listeners as being hand-made in the era of the RP2A03.

Jay McCarthy is an associate professor of computer science at the University of Massachusetts at Lowell and a member of the Racket development team. He completed a Ph.D. at Brown University in the Computer Science Department. His priorities are programming language expressiveness, formal verification, CS education, and family.

2.6 RAW (Seluk Artut, Alp Tuan)

The duo of Seluk Artut and Alp Tuan, as RAW, will perform using three laptops: one dedicated to the display of visuals generated with openFrameworks [8] and the other for Supercollider and Sonic Pi [1]. They will use a video switcher to switch between the two coding screens, the visuals, and a GoPro camera. Additional audio will be produced on an Arturia Minibrute, an analog synthesizer.

Seluk Artut lives in Istanbul, Turkey, where he spends much of his time philosophising human-technology relations. An author of four books in the past, his artistic activities are mainly focused on contemporary art practices based on technological embodiments. With an academic background in mathematics and sonic arts, he received his PhD on Philosophy of Media Communications from the European Graduate School in Switzerland.

Alp Tugan lives in Istanbul. Tugan is an interaction designer focusing on creative coding for his artistic productions. He has received his MA on Visual Communication Design from Sabanci University, Istanbul.

2.7 Francis Gene Shuman and Yullippe

Gene Shuman will present a visual performance with his system called Epimorphism, an art project intended to simulate video feedback. It is written in PureScript, a strict dialect of Haskell which compiles to Javascript and runs in web browsers supporting WebGL. Video feedback is a traditionally analog art form used since the 1960s to create recursive and self similar video animations. Despite feeling a lack of a particular domain-specific justification for using functional programming, Shuman argues that the use of functional programming for this project shows the inherent benefits of functional programming: faster and more stable development of software which is easier to reason about and to return to after significant developmental pauses.

Yullippe will accompany the visuals with a live electronic music performance of ambient and techno using a Moog Mother 32 synthesizer and DAW software.

Gene Shuman is a software engineer in the San Francisco area, where he moved in 2007 after studying mathematics, computer science, and electrical engineering at the Massachusetts Institute of Technology. In his spare time he is an amateur musician, aspiring artist, and world traveler. His current interests include the convergence of technology and the arts, meditation, and cats.

Yullippe (Yuri Urano) is a musician from Osaka. She has been performing as Yullippe since 2012. She has released two albums and performs frequently at electronic music events in Japan.

2.8 Atsushi Tadokoro

Atsushi Tadokoro will perform a piece called Synesthesic Stream, a live coding audiovisual performance using TidalCyles. All audio and visuals are generated in real-time through improvisation in code. In the work, the sounds gradually change, and the visuals follow the sounds according to the performer's "synesthesic" sense. The visuals are generated by generating control signals in TidalCycles which are sent via OSC using Tadokoro's own library to a video synthesizer which he has developed using openFrameworks.

Atsushi Tadokoro is a creative coder, artist, algorithmic audio visual improviser, and programmer. In 2010, he wrote a book "Be-

yond Interaction" which was the first openFrameworks handbook in the world. He is a lecturer in creative programming at Tama Art University in Tokyo, Japan and the Tokyo University of Arts.

References

[1] Aaron, S. and Blackwell, A.F. 2013. From sonic Pi to overtone: Creative musical experiences with domain-specific and functional languages. *Proceedings of the first ACM SIGPLAN workshop on Functional art, music, modeling & design* (2013), 35–46.

[2] Bell, R. 2011. An Interface for Realtime Music Using Interpreted Haskell. *Proceedings of LAC 2011* (Maynooth, Ireland, 2011).

[3] Drape, R. 2009. *Haskell supercollider, a tutorial*.

[4] Drape, R. 1998. Rsc3.

[5] Dybvig, R.K. 1996. *The Scheme Programming Language: ANSI Scheme*. Prentice Hall PTR.

[6] Freeman, P. 2016. PureScript.

[7] Jones, S.P. et al. 1993. The Glasgow Haskell compiler: A technical overview. *Proc. UK Joint Framework for Information Technology (JFIT) Technical Conference* (1993).

[8] Lieberman, Z. et al. 2009. *Openframeworks*.

[9] McCartney, J. 1996. SuperCollider.

[10] McLean, A. and Wiggins, G. 2010. Tidal - Pattern Language for the Live Coding of Music. *Proceedings of the 7th Sound and Music Computing conference* (2010).

[11] Sorensen, A. and Gardner, H. 2010. Programming with time: Cyber-physical programming with impromptu. *ACM Sigplan Notices* (2010), 822–834.

[12] 2002. *Haskell 98 Language and Libraries: The Revised Report*.