CPSC 290 Proposal

Extending Functional UI Library for Haskore

Dung Trung

January 20, 2009

1. Motivation:

Functional Reactive Programming (FRP) is a programming paradigm for reactive programming using the building blocks of functional programming to model hybrid systems containing both continuous time-varying behaviors and discrete events. FRP has been applied to a variety of systems, including computer animation [EH97], robotics [HCNP02], and real-time systems [WTH01].

In computer music, we want to expose to the programmer the notion of discrete notes, events, and continuous audio signals, as well as the transition and interaction between these entities. The hybrid and dynamic nature of the problem makes FRP a good choice for modeling computer music systems.

Haskore [HMGW96], a domain-specific language (DSL) built on top of Haskell, takes an algebraic approach to high-level music description and composition. Haskore has built-in functions to represent notes, rhythms, chords and many more. In Haskore a basic set of widgets is provided that are collectively referred to as the Graphical Music Interface [H08].

This interface has two levels of abstraction: At the user interface (UI) level, monadic interface is used to create graphical sliders, push-buttons, and so on for input, and textual displays and graphic images for output. In the final project of the course "Computer Music – Algorithmic and Heuristic Composition", I have contributed the virtual keyboard to Haskore widget library. The second level of abstraction of the GMI is the signal level. A signal is a time-varying quantity that nicely captures the behavior of many GUI widgets. In addition to these graphical widgets, the Haskore's UI level also provides an interface to standard Midi input and output devices.

However, Haskore GMI provides only a limited set of UI widgets, all written from scratch using SOE graphics [H00] built on top of OpenGL. Although the current approach is self-contained, simple, extensible and portable, in the long run, we would like to integrate with mainstream GUI frameworks so that we will have a richer set of widgets at our disposal, and avoiding having to re-implement complex controls, such as drop-down menus, plots, advanced text boxes and so on.

The first level of abstraction in Haskore is similar Phooey, a popular functional UI library for Haskell. Therefore we'd preserve Phooey-style monadic interface for wiring UI widgets with signals. One of the main challenges is that it is difficult to integrate Haskore with an external GUI package out there while still maintaining the signal level of abstraction. In addition, we need to allow a backward-compatibility to use old Haskore-specific widgets, such as the virtual keyboard and knob widgets. One approach is to embed old widgets in an OpenGL canvas on which such widgets can still be used rendered.

2. Proposal

This project aims to solve the various problems discussed above. In particular, we will extend the Graphical Music Interface (GMI) for Haskore by integrating it with a popular GUI framework, so as to create a richer set of GMI widgets. To do this we will:

1. Investigate the way Phooey uses wxHaskell to provide an FRP-like abstraction for GUI programming.
2. Adapt this technique to use our current implementation strategy for GMI, thus avoiding a potential space leak.
3. Create a new Application Programming Interface (API) for the GMI that is consistent with the principles underlying Functional Reactive Programming.
4. Devise a way to incorporate extensions -- such as widgets for a piano keyboard, guitar fret, controller knob, etc. -- into the overall framework, possibly using the Open GL canvas, but maintaining an API consistent with the above.
5. The overall goal of this project is to keep Haskore's GMI interface as efficient yet elegant as possible and avoid known problems with existing libraries, such as space leaks.

3. References

[E09] Eric Cheng. *A Purely Functional Interactive Computer Music Framework*

[EH97] Conal Elliott and Paul Hudak. *Functional reactive animation*. In International Conference on Functional Programming , 1997.

[HCNP02] Paul Hudak, Antony Courtney, Henrik Nilsson, and John Peterson. *Arrows, robots, and functional reactive programming*. In Johan Jeuring and Simon L. Peyton Jones, editors, Advanced Functional Programming , volume 2638 of Lecture Notes in Computer Science , pages 159–187. Springer, 2002.

[WTH01] Z. Wan, W. Taha, and P. Hudak. *Real-time FRP*. In International Conference on Functional Programming , Florence, Italy, September 2001

[HMGW96] Paul Hudak, Tom Makucevich, Syam Gadde, and Bo Whong. *Haskore music notation - an algebra of music*. Journal of Functional Programming, 6(3):465–483, 1996.

[H00] Paul Hudak. *The Haskell School of Expression: Learning Functional Programming through Multimedia*. Cambridge University Press, New York 2000

[H08] Paul Hudak. *The Haskell School of Music*. Yale University, 2008