Don't Forget the Laptop: Using Native Input Capabilities for Expressive Musical Control

Rebecca Fiebrink, Ge Wang, and Perry R. Cook†
Department of Computer Science †(also Music)
Princeton University
35 Olden St.
Princeton, NJ 08540
+01 (609) 258-8263
{ fiebrink, gewang, prc }@cs.princeton.edu

ABSTRACT
We draw on our experiences with the Princeton Laptop Orchestra to discuss novel uses of the laptop's native physical inputs for flexible and expressive control. We argue that instruments designed using these built-in inputs offer benefits over custom standalone controllers, particularly in certain group performance settings; creatively thinking about native capabilities can lead to interesting and unique new interfaces. We discuss a variety of example instruments that use the laptop's native capabilities and suggest avenues for future work. We also describe a new toolkit for rapidly experimenting with these capabilities.

Keywords
Mapping strategies. Laptop-based physical interfaces. Collaborative laptop performance.

1. MOTIVATION
One driving philosophy of the NIME community is that controller design greatly influences the sort of music one can make [1]. With respect to this, we are interested in developing control methods in specific settings (e.g., laptop ensembles), and in encouraging the community to consider all impacts of controller choice in scenarios presenting practical limitations. Customization of controller to musical task is desirable, yet so are availability, ease of use, development time, and portability. In a real-world environment, these needs must all be addressed to maximize musicality, efficiency, and fun. To achieve this end, one must think creatively about all inputs at one's disposal.

Music performed using laptops has a large and growing body of practitioners in both academic and popular realms. Recent proliferation of software tools (PD [8], SuperCollider [6], ChucK [10], also new uses for Perl [7], Python, etc.) has greatly reduced the barriers of expertise, time, and money required to create music, opening the door to hobbyists as well as extending possibilities for dedicated performance groups. Our experiences with the Princeton Laptop Orchestra (PLOrk) [9], an ensemble of laptop meta-instruments, highlighted several issues one confronts in creating live computer-mediated performances for such a group. Among these issues are how to foster musical expression in a variety of pieces, how to create pieces that musically engage the performers and audience, and how to support composers in developing pieces for the ensemble, in addition to all the practical concerns of maintaining an ensemble of laptops.

Drawing on these experiences, we hope to contribute to the discussion surrounding expressive and effective control interfaces for collaborative laptop performance in research, compositional, and informal contexts. In particular, we recognize that custom standalone music controllers can be highly useful, but experiences show they come with their set of hurdles. These include exacerbating the long set up/tear down time, complicating transportation, requiring expensive sensors or components and expertise in their construction and maintenance, and presenting steep learning curves to players. Furthermore, in ensembles such as PLOrk, many composers work with the players during rehearsals to develop their pieces and associated interfaces. Thus, rapid experimentation, familiarity with control interfaces, and reduced development and setup overhead are often essential to the successful crafting of a performance work. The central issue we address here is how to mitigate the problems custom controllers present for such an ensemble, while allowing expressive and flexible control and experimentation for a variety of pieces.

Fortunately, the innate capabilities of laptops themselves continue to present new opportunities for control. Devices such as accelerometers and cameras are now often built-in, and standard input methods such as keyboards and trackpads can be used in innovative ways (i.e., other than for executing commands, manipulating graphical interfaces, etc.). In the context of crafting instruments, the self-contained laptop has several advantages that alleviate some of the difficulties associated with custom controllers (e.g., cost, availability, portability). Laptops are ubiquitous, and it is easy to develop and distribute software compatible with built-in components. They are easy to transport and require no special maintenance and setup overhead compared with many customized standalone controllers.

Although input devices such as keyboards and trackpads are simple and not physically configurable, software flexibility presents unexplored possibilities for using these devices in new

and musically interesting ways. Additionally, the aforementioned benefits of relying solely on built-in components contribute to the smooth functioning and creative well-being of a laptop performance ensemble. Therefore, we begin by positing that laptops themselves merit the continued attention of musical interface designers and researchers. We present examples of using traditional and non-traditional laptop inputs in new ways and suggest additional means of exploiting laptop capabilities. We also describe a new lightweight toolkit for quickly experimenting with and utilizing several of the native input capabilities of the laptop.

2. BACKGROUND

Laptop music performance dates back as far as laptops themselves, but it especially began to take off in the 1990's. Smoky clubs from Tokyo to Berlin, LA to New York started to host nights dedicated to noise, glitch, infrasound, and other new electro-acoustic genres afforded by the new powerful portable computers. Some used commercial sequencing software, while others opted for more general purpose languages such as MAX/MSP or SuperCollider. For those inclined to make more traditional music, software programs such as ReBirth (emulating the Roland 606, 808, 303 and other drum/bass machines), and Reaktor (emulating a modular synthesizer) became available [4,12]. Recently the practice of "live coding" has become more popular, in which the performer(s) actually program the computer live, often projecting the screen [11].

Many laptop performers have exclusively used the capabilities inherent in laptops for music and other control tasks, even though they may not have discussed the choice to forego a more customized control solution. Obviously, keyboard and mouse are nearly always used to control GUI's such as patches, or possibly to write code. In these cases, keyboard and mouse inputs translate into onscreen operations, which then influence the music. We, on the other hand, are interested in the use of such devices wherein each key press, mouse gesture, or other physical interaction is *directly* musically meaningful.

For example, an innovative, though non-musical, use of a built-in laptop capability is the "SmackBook," where users can perform user interface operations by physically hitting or tilting the laptop. Popularized in an internet video [2], this clever "hack" uses the built-in sudden motion sensor designed to protect the hard drive. We hope to transport this type of creative resourcefulness into the musical domain.

3. MAPPING STRATEGIES

In the following section, we describe several native laptop input capabilities with case studies of their use in PLOrk instrument design and performance. Each instrument is crafted around the physical nature of the input and allows the performer to rely on the inherent physicality of the device (and its mappings), without the need for onscreen interfaces. Much of the code used in these pieces is included in our toolkit, which is discussed in Section 5.

3.1 Keyboard

While the decidedly discrete nature of the laptop keyboard makes it impractical for some tasks, its form factor is optimized for small, fast, and precise finger movements, and musical mappings might leverage the performer's existing typing skills. For this reason, the keyboard can be a natural, if simple, musical controller. In Wang's *CliX*, performed by PLOrk, human operators type to trigger sounds, which are synthesized, synchronized, and spatialized by their laptops. Every key on the computer keyboard (upper- and lower-case letters, numbers, and symbols) is mapped to a distinct pitch using the key's ASCII representation, and when pressed, the laptop emits a clicking sound that is synchronized through the ensemble to a common pulse. A human conductor coordinates frequency range, texture, global spatialization, and timing.

The mapping is easily understood by players, who can immediately begin to make sounds without practice. The ASCII-based layout makes it difficult to play melodies but is sufficient for selecting relative pitch regions based on the alphabet (for example, letters 'U' through 'Z' result in lower pitches than 'a' through 'd').

In another set of pieces, keys are mapped to pitches in a fretboard-like configuration, so notes and chords can be played with one hand with minimal hand displacement (see Figure 1), leaving the other hand to operate a different controller. This particular mapping was first used in Wang's *Crystalis* and later extended in Fiebrink, Wang, and Cook's *Joy of Chant*.

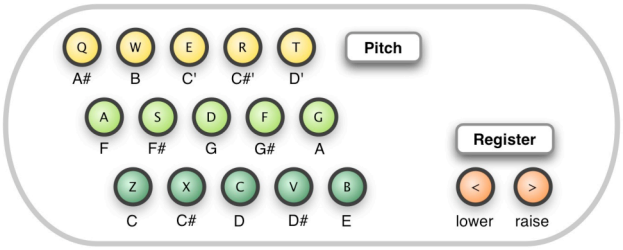

Figure 1. Fret-based pitch selection.

Performers of *Crystalis* follow the conductor to adjust pitch, density, and volume, similar to *CliX*, while controlling other parameters using the trackpad (discussed in the next section). In contrast, *Joy of Chant* requires performers to use one hand to select specific pitches in unison from a score, while the other hand controls singing synthesis parameters via a standard joystick. *Joy of Chant* extends the pitch selection keys rightward to include the entire keyboard. Mappings of both pieces extend the pitch range by providing means to shift registers in octave increments. In both cases, the performers found the keyboard interface easy to use and were able to perform after a few rehearsals.

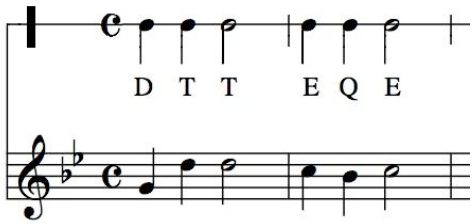

Figure 2. *Joy of Chant* score (top) with traditionally transcribed equivalent (bottom).

Compared to the ASCII-based mapping, the fret-like mapping makes it easier to play melodies; players need only remember the physical locations of pitches and not of the letters. The letters can

be notated in a score to help players learn the music and the mapping (see Figure 2).

3.2 Trackpad

Looking past its use as pointing device, the ability of the trackpad to track two-dimensional motion offers a wide array of mapping strategies. One of the most heavily researched HCI devices [14], the modern trackpad offers fine-grained, low-latency sensitivity with tactile and visual feedback.

In *Crystalis*, players "bow" the trackpad by varying finger location and speed to inject "energy" into a synthesis model, in tandem with keyboard pitch control. This mapping involves capturing finger motion (as relative *x* and *y* position updated at interactive rates), which is low-pass filtered (2nd order Butterworth with 10 Hz cutoff). This signal is then passed into a leaky integrator (one-pole low-pass filter) that generates an envelope with a smooth attack and gradual release, which is in turn multiplied with the source signal (wind-like sounds or a banded waveguide model). Moving the fingers quickly tends to result in louder and more energetic sounds. Bowing in different directions places sound into particular audio channels. Physical finger patterns directly correspond to audio/spatial patterns (see Figure 3). For example, making a circular pattern on the trackpad will move the sound smoothly around the output channels. The tight coupling between finger location and spatialization and between player effort and sound energy make this a natural mapping.

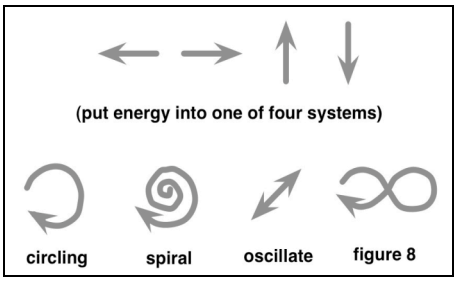

Figure 3. Trackpad bowing motions.

3.3 "Smack-sensing"

A less commonly used input capability is the accelerometer-based motion sensor found in many laptops. In Fiebrink's *Smacking Music*, PowerBook laptops are used as "acoustic" percussive instruments to perform Steve Reich's *Clapping Music*. Performers hit their laptops with hands or other objects to produce the auditory component of the piece. The built-in sudden motion sensor is polled by a software application, and motion events surpassing a certain amplitude threshold are registered as "smacks." In response to each smack, the laptop display is modified, producing a synchronous visual accompaniment.

The piece is performed with any number of people, separated into two groups, one for each part of the original score. Laptop screens face the audience, who observes both smacking gestures and laptop screens. Performers are encouraged to hit the laptop wherever and however they like. The laptop motion sensor is suitably sensitive that little force is required (e.g., tapping the laptop base with a pen). Fortuitously, the motion sensor continues to perform its intended role of protecting the laptop hard drive from harm throughout the piece, minimizing the risk posed to the computer by the performer. The experiences of performing and observing this piece are quite novel and entertaining.

The sudden motion sensor output provides absolute tilt information in three axes dozens of times per second. Other physical stimuli, such as shaking or tilting, can also easily be captured, conditioned, and used as control parameters.

3.4 Laptop Microphone and Speaker

The laptop's integrated microphone and speakers, while far below the quality of those typically employed in laptop performance, can themselves be used as novel input/output devices. Matt Hoffman's *Breathalyzer?* requires performers to blow directly into the laptop's microphone. The sound of the piece is band-pass filtered computer generated noise; when the noisy input of the performer's breath is detected, the center frequency of the bandpass filter is changed. The piece takes form as performers influence their filters as a coordinated ensemble.

In this piece, the microphone senses the presence or absence of breath. The use of microphone input can easily be extended to track continuous variables, such as the amplitude envelope of the breath, for other mappings. The quality of the laptop microphone is satisfactory for capturing this coarse control data.

While the input methods addressed in this section can be used with external speakers to produce high-quality sound output, it is also interesting to consider their use in spontaneous, highly portable settings. In fact, *Breathalyzer?* is one of a series of PLOrk pieces, *Small Sound Sketches*, composed to use the laptop speakers (and humans) as the sole means of output (this is as close as we can get to "PLOrk Unplugged"). *Small Sound Sketches* is an example of how working within laptops' inherent constraints, in pursuit of extreme portability in this case, can produce novel and interesting results.

4. OTHER STRATEGIES

4.1 Webcams

The video camera has been used as a musical input device in the past, for example in the Mouthesizer [5]. As many laptops begin to be shipped with built-in webcams, video and photo input capabilities are increasingly available for use in laptop music. Future work for our toolkit includes integrating webcam functionalities such as raw video capture and basic image analysis, so users can easily integrate live input, such as the Mouthesizer, into any performances.

4.2 Networking

Today's laptop comes standard with capabilities for easily creating ad hoc wireless networks, without the need for extra hardware. Furthermore, developing software to communicate with other laptops on a network is quite straightforward, particularly when using established protocols such as OpenSound Control [13]. Several PLOrk pieces have used networking as an integral component. In *CliX*, for example, a machine "conductor" synchronizes and quantizes the sounds triggered by each player's keyboard by emitting periodic pulses via OSC, leveraging the computer to augment the degree of control offered by the keyboard. In Fiebrink and Wang's *PLOrk Beat Science*, five networked laptops act as a distributed sound bank, and a player at one machine can trigger sounds on other machines to create spatial patterns.

4.3 The Kitchen Sink

Even less obvious channels for communicating information to a laptop might be used for musical control. Other increasingly standard laptop features include Bluetooth and remote control (e.g., MacBook and Dell Inspiron series). One might devise pieces to take advantage of even the most trivial laptop controls, such as buttons for power, volume, brightness, etc. Operating system-specific tools such as AppleScript can also be used effectively to access and control basic system features. While latency, bandwidth, and other issues may limit the usefulness of such controllers, we believe that mundane features may still have musically interesting applications.

Laptop technology may soon grow to include yet more varied and promising interfaces for control. A 2005 Apple patent application describes a "mechanical overlay" touch-sensitive interface, which would integrate into the laptop hot-swappable mechanical controllers, such as knobs, sliders, joysticks, and piano-like keyboards [3]. If such technologies come to be standard, these controllers would open up even more opportunities for laptop-contained control with the aforementioned benefits of portability, low maintenance overhead, and ubiquity. Musical interface designers should stay informed of such developments with an eye toward their obvious and non-obvious potential uses in music.

5. A NEW TOOLKIT

We have assembled a publicly available toolkit, the Small Musically Expressive Laptop Toolkit (SMELT), to facilitate rapid development and experimentation using some of the control capabilities mentioned above (http://smelt.cs.princeton.edu/). This toolkit contains a collection of ready-to-use source code modules and examples. Many of the tools arise out of our previously discussed work with PLOrk, and are summarized in Table 1 below.

Table 1: Toolkit components

Input	Description	Examples
Keyboard	ASCII and fret-like pitch selection	*CliX*, *Crystalis*, *Joy of Chant*
Trackpad	mouse and trackpad bowing	*Crystalis*
Motion	motion sensing and signal conditioning code, user API	*Smacking Music,* formant control
Mic	breath control	*Breathalyzer?*, envelope follower

We hope that releasing this toolkit will encourage other performers, composers, and researchers to similarly make available their code for capturing laptop inputs for novel musical expression. While current SMELT examples are in ChucK, we welcome contributions in other languages. Our vision is that these tools might join the standard palette used to craft collaborative laptop music performance. A critical mass of ubiquitous, easy-to-use code can encourage willing experimenters to make more music together with their laptops, while continuing to ponder and refine the use of laptop inputs in their music-making.

6. CONCLUDING REMARKS

Custom interfaces play a necessary role in facilitating expressivity and creativity in performance of new music. At the same time, certain rehearsal, composition, and performance paradigms can benefit from the low overhead, ease of use, and availability of the native laptop input capabilities. Constructing instruments that creatively exploit these capabilities can lead to interesting musical possibilities. In our experiences with the Princeton Laptop Orchestra, we have found the use of native laptop inputs to support the development of new compositions in a group setting, and to be effective in performing a variety of compositions. We believe that laptop controls are not only useful in PLOrk-like laptop ensembles and electronic chamber music settings, but they also encourage spontaneous and informal musical collaboration. Both paradigms can benefit from reducing barriers of cost, overhead, learning curve, etc. while preserving a variety of control options. The laptop is a popular and evolving instrument that can be played anywhere, anytime, by anyone. We hope that our work, and that of others interested in musical interfaces, can increase the expressive capabilities of laptops by calling attention to and writing code in support of the laptop's natural capabilities for novel musical ends.

7. ACKNOWLEDGMENTS

Our thanks to the Princeton Laptop Orchestra, Dan Trueman, Scott Smallwood, Matt Hoffman, and Spencer Salazar.

8. REFERENCES

[1] Cook, P. R. 2001. Principles for designing computer music controllers. *ACM CHI Workshop in New Interfaces for Muiscal Expression*.

[2] Ellingsen, E. 2006. SmackBook Pro. http://blog.medallia.com/2006/05/smacbook_pro.html.

[3] Huppi, B. Q. 2005. Mechanical overlay. U.S. Patent Application DN/20060256090. Apple Computer, Inc.

[4] Loubet, E. 2000. Laptop performers, compact disc designers, and no-beat techno artists in Japan: Music from nowhere. *Computer Music Journal* 24(4): 19–32.

[5] Lyons, M. J., M. Haehnel, and N. Tetsutani. 2001. The Mouthesizer: A facial gesture musical interface. *Conference Abstracts, SIGGRAPH*.

[6] McCartney, J. 1996. SuperCollider: A new real-time synthesis language. *Proc. ICMC*.

[7] McLean, A. 2004. Hacking Perl in nightclubs. O'Reilly Media, Inc. http://www.perl.com/pub/a/2004/08/31/livecode.html.

[8] Puckett, M. 1996. Pure Data. *Proc. ICMC*.

[9] Trueman, D., P. R. Cook, S. Smallwood, and G. Wang. 2006. PLOrk: Princeton laptop orchestra, year 1. *Proc. ICMC*.

[10] Wang G., and P. R. Cook. 2003. ChucK: A concurrent, on-the-fly audio programming language. *Proc. ICMC*.

[11] Wang, G., and P. R. Cook. 2004. On-the-fly programming: Using code as an expressive musical instrument. *Proc. NIME*.

[12] Weidenbaum, M. 2006. Serial port: A brief history of laptop music. *The Web Magazine of the American Music Center*. http://www.newmusicbox.org/article.nmbx?id=4659.

[13] Wright, M., and A. Freed. 1997. Open Sound Control: A new protocol for communicating with sound synthesizers. *Proc. ICMC*.

[14] Zhai, S. 2004. Mouse. In *Encyclopedia of human-computer interaction*, edited by W. S. Bainbridge: Berkshire Publishing Group.