About This Issue

The previous issue of *Computer Music Journal* offered a look at the practice of live coding on laptop computers, approximately ten years after the inception of live coding. Similarly, this issue's first article—the only one reflected by the topic on the issue's front cover—uses the present as a vantage point for considering the recent practice of using mobile phones as musical instruments. The author, Ge Wang, describes the iPhone application Ocarina, which uses his programming language ChucK for sound synthesis. Introduced in 2008 and downloaded by millions of users, Ocarina was one of the earliest musical applications for the iPhone, but Wang's article dutifully credits its predecessors on other platforms. Ocarina uses the iPhone's microphone to detect the player's breath, effectively turning the device into a wind instrument. The multi-touch screen permits ocarina-like fingering along with non-ocarina-like visualization schemes. The phone's accelerometers enable expressive vibrato, and its global positioning system facilitates social interaction.

One approach to creating new musical interfaces is to use the capabilities of an existing device in a nontraditional way, as is the case with Ocarina's use of the iPhone's microphone. Another approach is to design a new custom piece of hardware, which can be a challenging and expensive proposition. The article by Rodolphe Koehly and colleagues addresses the challenge by considering the use of an inexpensive material, paper containing a conductive pigment, as a force sensor that can be incorporated into musical controllers. Paper has advantages besides low cost: it can easily be cut as needed and shaped (for example, in the form of a curved surface), and sensors made of paper are efficient and have a low environmental impact. The authors describe controllers that have been implemented using paper sensors, including stringed instruments, drumsticks, and a glove, among others, as well as larger matrices of paper sensors applied to a floor or wall.

The next pair of articles, by the well-known computer music researcher Roger Dannenberg and his colleagues, considers the challenges for a computer that must perform along with a human ensemble. Focusing on popular music, the authors assume that the computer will synchronize to steady beats in the music being performed by its human bandmates. This task of alignment is complicated by the fact that the human musicians might deviate from a predefined musical "score"—for example, by deciding midstream to repeat a chorus, omit a verse, or extend a vamp. Thus the problem is a superset of score following, a field that Dannenberg himself helped to introduce three decades ago and that assumes a completely fixed musical score. In their first article, the authors propose a taxonomy of "human–computer music performance" scenarios and lay out a set of predictions for future features and requirements of such systems. They then describe a reference architecture and a specific implementation. The latter can be experienced through a video recording of an actual musical performance by a jazz ensemble augmented by a virtual string orchestra.

The authors' second article delves into some related details. These include: a synchronization technique that accounts for latency due to communication delays and audio buffering; a strategy for mapping from a pre-performance "static" score to a performance-time "dynamic" score that may differ from the static score in ways alluded to earlier; and finally, what the authors call an "active score": an on-screen, conventionally notated score that depicts the computer's current temporal position and lets a human performer point to a new position to which the computer should move.

The issue's final article is an extended version of the manuscript that won the Best Paper award at the 2013 International Computer Music Conference (ICMC) in Perth, Australia. Israel Neuman's groundbreaking ICMC paper applied generative grammars in an area to which computer scientists have paid scant attention: *musique concrète*. Neuman took Pierre Schaeffer's summary table of sound classification, the *Tableau Récapitulatif de la Typologie* (TARTYP), and used it to derive grammars that serve as the basis for new musical structures in interactive composition. Java classes embedded in Max/MSP or Pure Data can process sound files, or live sounds, to endow them with the acoustic attributes of the TARTYP classification and

doi:10.1162/COMJ_e_00235

Front cover. This figure from the article by Ge Wang shows frames from various YouTube videos created by Ocarina users.

Back cover. The top two figures, from the article by Koehly et al., depict performers whose musical instruments incorporate sensors made of paper. On the bottom, three iPhone screen images illustrate Ocarina's visualization of the act of listening to other users around the world.

to organize the sounds in real time according to the rules of a particular grammar. Neuman's *CMJ* article extends this work by describing another project of his that similarly allows composers to generate musical structures in real time. This newer project organizes pitches into structures using Klumpenhouwer networks, which are normally an analytical tool for identifying pitch relationships in a pre-existing composition.

The reviews in this issue assess a CD of compositions by Cuban electroacoustic pioneer Juan Blanco, a conference at Cambridge University on performance studies, and Eric Lyon's textbook on how to write audio objects in the C programming language for use in the graphical music environments Max/MSP and Pure Data (Pd). The Products of Interest section describes a number of recent audio interfaces, microphones, keyboard instruments, controllers, synthesis kits, and plug-ins.

Prospective authors should note that in 2013 *Computer Music Journal* introduced a new online system for submitting manuscripts. For details, please see the instructions to contributors at the end of this issue.