A Mobile Wireless Augmented Guitar

N. Bouillot,
M. Wozniewski
McGill University Centre for Intelligent Machines
Montréal, Québec, Canada
{nicolas,mikewoz}
@cim.mcgill.ca

Z. Settel
Université de Montréal
Montréal, Québec, Canada
zs@sympatico.ca

J. R. Cooperstock
McGill University Centre for Intelligent Machines
Montréal, Québec, Canada
jer@cim.mcgill.ca

ABSTRACT

We present the design of a mobile augmented guitar based on traditional playing, combined with gesture-based continuous control of audio processing. Remote sound processing is enabled through our dynamically reconfigurable low-latency high-fidelity audio streaming protocol, running on a mobile wearable platform. Initial results demonstrate the ability to stream the audio and sensor data over IEEE 802.11 to a server, which then processes this data and outputs the resulting sound, all within a sufficiently low delay as required for mobile multimodal performance.

1. INTRODUCTION

Electronic instrument augmentation can allow an already skilled musician to exploit gestures to gain extra control over the resulting sound. Typically, additional sensors are attached to the instrument or body in order to acquire gesture-related data. Existing augmented instruments often transmit sensor data using serial [?], USB [?] or Ethernet interfaces [?]. Unfortunately, such wired interfaces can reduce artists' mobility during performance. Wireless sensor platforms have been proposed to overcome this situation. Implementations include the WiseBox [?], and Bluetooth Arduino [?], among others. However, these interfaces do not support wireless transmission of audio signals, instead requiring cables, often analog, to send audio to the processing machine.

In related work, we proposed methods for high quality wireless audio transmission [?], which alleviate the need for cables entirely. With the availability of a wireless capability for audio transmission, we were motivated to explore the potential of a mobile platform to support performance with an augmented musical instrument, in this case, a guitar. Using WiFi potentially lets the musician walk around a much larger space can also take advantage of remote processing capabilities of computers over IP rather than requiring all audio gear to be situated locally avoids interference from common devices such as cellular telephone. In term of musical interaction, our intent was to demonstrate how such an instrument could provide for musical expression that would otherwise require an inordinate investment of training, or additional musicians. Examples include self-accompaniment and self-duo. The former is typical of Indian classical music instruments, which may provide harmonic resonance from a basic note or the more interesting gesture-based Satara double flute from Rajastan, in which the absence of holes in the second flute result in a steady drone. Self-duo can be seen in the playing technique developed by jazz musician Roland Kirk, in which a single musician plays two (or more) reeds at once. However, simultaneous management of multiple instruments seriously reduces the range of gestures available on each one. Playing techniques require considerable practice, for example, to regulate circular breathing or coordinate new hand and body positions.

Our particular implementation is based on a small form-factor computer that runs a fixed-point version of Pure Data [?]. Leveraging the Bluetooth communication capability of the Gumstix, we can read accelerometer values, position information (of IR sources), and button presses of a Nintendo Wii controller affixed to the guitar headstock. Interaction is based on traditional playing technique, augmented by a set of simple gestures for invoking added functionality, such as sample recording and looped playback. These features can be used to simulate simultaneous performance with multiple instruments, as described above. Gesture data and the resulting audio must be communicated with sufficiently low delay to allow the performer to understand the relationship between the gesture and corresponding sound. We thus developed an dynamically reconfigurable low latency streaming engine used in this implementation.

The remainder of this paper is organized as follows. Section 2 describes hardware and software components of our platform, keeping latency issues in mind. Section 3 presents our mobile augmented guitar and its main feature: the gesture-based phase vocoder control using real-time sampling. Mapping is also described and discussed. Before concluding, various applications enabled by our platform are explored.

2. WIRELESS PLATFORM

The mobile platform architecture is presented in Figure 1. The Gumstix captures audio data through its input jack and obtains gesture input and button-press information from the Wii controller over Bluetooth. Both the audio samples and sensor data are then transmitted by IEEE 802.11g (WiFi) to a laptop computer, where signal processing functions are performed. Open Sound Control (OSC) is used as the protocol for control data.

For wireless transmission of audio data, we developed a dynamically reconfigure low-latency, uncompressed audio streaming protocol[1] as a Pure Data external for both the

[1] It is worth noting that the industry is moving toward support for low-latency transmission of lossless audio over Blue-

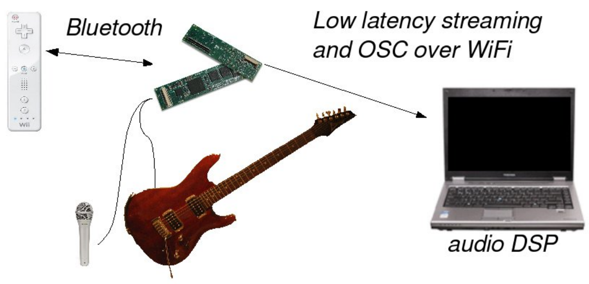


Figure 1: Mobile wireless platform architecture

Gumstix and the DSP machine (laptop).

The remainder of this section discusses our architecture, the gestural interface, and explores issues of computational requirements and low-latency streaming as necessary for audio feedback.

2.1 Gumstix and "Pure Data anywhere"

Gumstix computers [?] can be extended with various expansion boards for communication, I/O, and memory. Our platform is composed of a Gumstix main board (Verdex XM4-bt with Bluetooth) with two expansion boards for audio and WiFi. Figure 2 shows the Gumstix, a portable USB power pack, and the two antennas needed for Bluetooth and WiFi. A soundcard provides mini-jack input and output.

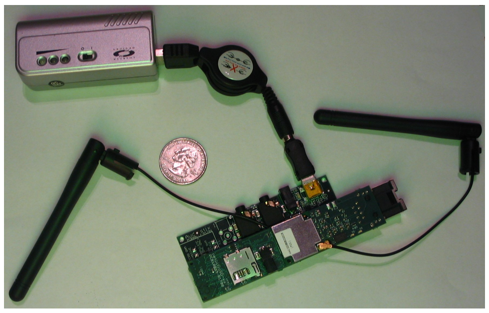


Figure 2: Gumstix size relative to a quarter

The software development for arm-based 400MHz Gumstix processors (Marvell's PXA270) requires a cross-compilation tool-chain to generate binaries. Floating point computations are allowed, but computed through software emulation, which means that Gumstix DSP code must be computed using fixed-point operations. For local Gumstix processing, we use Pure Data anywhere (PDa)[?], a rewritten fixed-point Pure Data (Pd) version[2]. This allows us to cross-compile existing externals such as OSC, but also to develop platform-related software such as real-time dynamically reconfigurable low-latency streaming (see Section 2.4) as well as sensor acquisition and filtering. Our software, along with PDa, easily fits within the base 15MB of memory supplied with the Gumstix motherboard.

2.2 Controller interface

The Wii remote provides a number of powerful capabilities in a single unit that greatly simplify prototyping where both accurate spatial positioning (e.g., temporal sweeping through a sound file) and mode selection (e.g., by button press or simple gesture) are required. While similar functionality could be obtained by a combination of other less "game-oriented" devices, we chose the Wii so that we could concentrate on the higher level issues involving musical expression rather than devote unnecessary resources to hardware concerns.

2.3 Remote sound processing

We first attempted to perform digital signal processing on the Gumstix as an autonomous unit. To provide an approximate reference of CPU demand for audio processing (beyond the stereo audio I/O overhead), we were able to run a maximum of one medium quality mono in/stereo out reverb module, *zverb* [?], consisting of the following units and their interconnections: nine 1-tap early reflection delay lines, one 4-tap feedback delay with quad circulator matrix, four low-pass filters, four interpolated multiple gain scalers, and several addition and multiplication units. Attempts to render multiple audio effects while performing low-latency streaming and sensor processing quickly overwhelmed the limits of the 400MHz Gumstix processor.

We have no doubt that future small form-factor computers will prove capable for more complex tasks. However, given current limitations, we opted, instead, to prototype using a remote sound processing server. The resulting implementation described here is thus a first step toward autonomous, augmented musical interaction.

2.4 Low-latency data streaming

Various delays are involved in the communication of audio during streaming. These can be broken down into the packetization delay, d_p, corresponding to the time required to fill a packet with data samples for transmission, and network delay, d_n, which varies according to network load and results in jitter at the receiver. Received data is typically held in a playback buffer, designed to maintain a constant latency and thus mask the effects of jitter and late arriving packets.

Packetization delay is calculated as:

$$d_p = n_{sp}/f \text{seconds} \qquad (1)$$

where n_{sp} is the number of samples per packet and f is the audio sampling frequency. Thus, we can affect d_p by varying the number of samples per packet, or changing the audio sampling frequency. During experiments, we set f to 44.1kHz.

In order to minimize latency, we developed a prototype dynamically reconfigurable transmission protocol, *nstream*, which supports both unicast and multicast communications. It permits dynamic adjustment of sender throughput by switching between different levels of PCM quantization (8 bit, 16 bit and 32 bit) and number of samples per packet (n_{sp}), as well as receiver buffer size, as suitable for low-latency, high quality audio streaming over wireless networks. This protocol is developed as a Pd external and allows interoperability between (the fixed-point) PDa and (the floating point) Pd.

Best results were obtained by streaming audio using n_{sp}= 64 with a receiver buffer of two packets, which corresponds to (64/44.1) x 3 = 4.35ms. We measured average `ping` time of 2ms, suggesting a network delay of approximately 6ms.

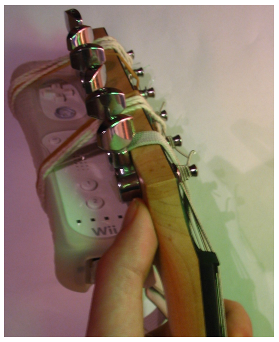
The Wii controller transmits sensor data at a rate that could be detrimental to the performance of a WiFi network, especially when already loaded with uncompressed audio. As an example, raw camera data is transmitted at a rate of 197Hz. Therefore, the Gumstix is used to filter such data and transmit control events only when relevant to the augmented guitar.

tooth, as evident by Qualcomm's acquisition of the Open Interface Soundabout audio codec.

[2]This is based on Pd v0.37 and was motivated for use on PDA devices.

3. AUGMENTED GUITAR AND MAPPING

The development of a mobile augmented guitar involves several issues. One challenge is the delay between acquisition of gestures and related feedback. Wireless audio streaming must be sufficiently fast to allow for real-time interaction and control. The 6ms of network delay, combined with minimal OSC overhead, was regarded as sufficient during experimentation.


(a) Wiimote fixed on headstock

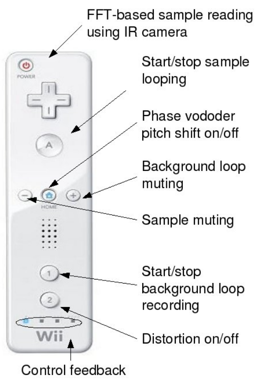

FFT-based sample reading using IR camera
Start/stop sample looping
Phase vododer pitch shift on/off
Background loop muting
Sample muting
Start/stop background loop recording
Distortion on/off
Control feedback
(b) Controls

Figure 3: Sensors and controls

To maximize performer mobility, all augmented functionality can be invoked through gestures made with the guitar and sensed by the attached Wii controller (see Figure 3(a)). These gestures include simple jerking motions and pointing with the neck of the guitar. Additionally, buttons are easily accessible for changing higher-level parameters. Musicians can start/stop the recording of an audio sample from the instrument using a quick flick of the guitar, avoiding the need for non-mobile controllers, such as a traditional pedal. Recorded samples can be played back using the infrared sensor, which controls the position of a play head. An FFT-based phase vocoder maintains the pitch of the original recorded material. An additional background loop can also be recorded in separate memory space. Together, these techniques allow for gesture-based processing to accompany traditional guitar playing.

3.1 Gesture-based phase vocoder

Accelerometers are used to detect fast movements performed by the musician, which are mapped to control the recording of musical samples. Once a sample has been stored, the infrared sensor on the Wii controller is used to capture (x,y) coordinates of an IR light source (e.g., the sensor bar provided with the Nintendo Wii system). This is used to control a phase vocoder based on a Pd patch by Miller Puckette. The vocoder allows for selective continuous listening to a small portion of a sound [?]. As seen in Figure 4, the x-axis is mapped to a time index in the current stored sample. A small window around that index is analyzed with an FFT to determine the spectral content of the location, allowing an inverse FFT to generate a time-stretched sound with equivalent pitch. The y-axis controls the pitch of the resulting sound at any location.

When the pitch-shifting functionality is enabled, both x and y axes coordinates are used to manipulate sound, allowing the musician to control playback with two dimensional movements, such as curved trajectories with the headstock or dance-like gestures. In the case that only time-indexing is desired, then the pitch-shifting feature can be disabled with a button press. Interaction thus becomes constrained to the x-axis, allowing the performer to play desired sections of the material with horizontal motions. This is interesting when harmonic progressions have been recorded, allowing musicians to explore tonal or atonal accompaniment, as described in Section 4.

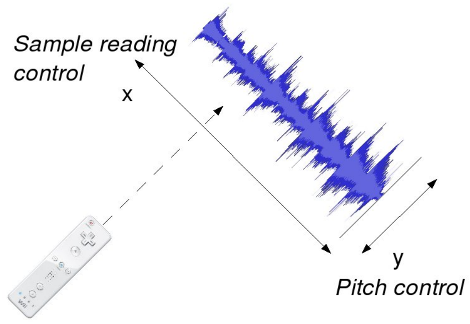


Figure 4: Gesture-based sample reading

Note that the gestures we support in the current prototype (jerk and pointing along the x or y axes) are extremely simple to recognize in that we need only apply a peak detector to determine the jerk and can use the blob detection functionality of the IR camera to provide accurate position information.

3.2 Controls

In addition to the phase vocoder described above, several additional features can be accessed using the Wii controller. The full list of mappings, including distortion, muting, and background loop control is illustrated in Figure 3(b). The signal recorded into the background loop contains not only the direct guitar signal, but also the result of applied distortion and the output of the phase vocoder. It should also be noted that there is an additional play control for looping the sample used by the phase vocoder instead of using IR time indexing.

The augmented guitar has four features that can be toggled on and off, including background loop recording, distortion, phase-vocoder sample recording and phase-vocoder pitch shift. The states of these features are shown by the four LEDs on the Wii controller. This provides feedback that is easily visible when the musician glances at the headstock, and is particularly useful when using the accelerometer-based activator since detection errors occasionally occur. Activation might not be detected when performer movements are too slow, whereas long motions can cause two detections sequentially, causing the function to be toggled on and off.

While accelerometers and IR sensors can easily be used during play, buttons may require gestures that disable guitar playing during the time taken to push the button. This leads us to consider the spatial layout, and the corresponding feature to which each button is mapped. As seen in Figure 3(a), buttons 1 and 2 can be pressed easily by the thumb when the guitarist's left hand is not using the fingerboard. This allows the guitarist to control background loop recording and distortion activation without changing hand position on the neck. To press other buttons, the hand must leave the fingerboard, so we assign those buttons to less frequently used functionality. Since the phase vocoder can be used without simultaneously playing the guitar, keeping the hand on the fingerboard is unnecessary, and buttons can become an integral part of the interaction. Future investigations will include other mappings and other kinds of sensors to enable or disable functionality. Important factors are mobility and ease of use during play.

4. MUSICAL PRACTICE

Several body movements are easily achievable by the musicians while playing the guitar. For example the neck can be moved horizontally and vertically while the musicians can walk, run and jump in any direction. The Wii controller, attached to the guitar's headstock, easily allows for the detection of neck movements; it provides control of the augmented guitar features while the instrument is being played simultaneously. This allows a musician to experiment with various forms of self interaction within solo or group performance.[3] Since performers are able to move around within the WiFi 802.11g range (100 meters), they may interact with the surrounding environment in additional ways. For example, a composer can arrange the space with ambient light or moving objects, with which the performer can interact. Multi-user interactions can also include IR LEDs or reflectors worn to augment group interaction.

In terms of musical dialogues, the augmented system provides some interesting techniques for an individual performer that usually cannot be accomplished with a single instrument. We consider the possibility of three such dialogues:

- *Questions and answers*: Samples can be recorded while playing, and then pointing the guitar in a particular direction replays the melody or sound in a different order and speed.

- *Self-duo*: The same technique as *questions and answers*, but pointing and simultaneously playing the guitar allows one to perform counterpoint-like interactions.

- *Self-accompaniment*: Recorded chords can be played back using dance-like movements while simultaneously playing guitar.

The dialogues that allow for playing two melodies simultaneously can be interesting in terms of tonal or atonal experimentation. Particularly, when pitch shift is disabled, a musician can record a progression of chords or a melody that can be used as the sample for the phase vocoder. The musician can then experiment with various guitar-played harmonies while triggering sounds from the recording using body movements.

5. CONCLUSIONS

We presented the design and experiment of a mobile wireless augmented guitar, in which powerful control mechanisms allow the user to control sonic events through simple gestures while simultaneously playing the instrument. Augmentation includes sample recording, looping, distortion and a gesture-based phase vocoding sample reader. We described our experience with the initial prototype, as well as new musical practices, including gesture-based self-duo and self-accompaniment that it supports.

The benefit of mobility, such as that provided by our small form-factor wireless system, is that its features can be used in classrooms or during rehearsals and do not depend on the technology available at a particular venue. However, the computational demands for signal processing necessitate, at present, the use of a remote computer for generating the audio output. In such a context, remotely computed audio raises the crucial issue of feedback latency. Fortunately, experimental results demonstrate that our dynamically reconfigurable audio streaming protocol satisfies both the timing and fidelity requirements for the demands of musical performance. The gesture-controlled phase vocoding features of our system suggest some interesting application possibilities, including pedagogical, note matching for counterpoint techniques, and musical transcription, where sample-part selection is controlled by the user's movements.

Our experiences with this platform lead us to believe that in the near future, a more capable mobile platform will extend the range of pedagogical and artistic practice. This may open up a new range of interaction possibilities in such creative areas as music, theater and dance.

[3]Demo videos are available at
`http://cim.mcgill.ca/~nicolas/NIME2008/`

6. ACKNOWLEDGMENTS

The authors wish to acknowledge the generous support of NSERC and Canada Council for the Arts, which have funded the research and artistic development described in this paper through their New Media Initiative.

7. REFERENCES

[1] nSlam: TOT [Territoires Ouverts - Open Territories] Website. http://tot.sat.qc.ca/logiciels_nslam.html.

[2] Arduino. http://www.arduino.cc.

[3] F. Bevilacqua, F. Guédy, N. Schnell, E. Fléty, and N. Leroy. Wireless sensor interface and gesture-follower for music pedagogy. In *NIME '07: Proceedings of the 7th international Conference on New Interfaces for Musical Expression*, pages 124–129, New York, NY, USA, 2007. ACM.

[4] G. Geiger. PDa: Real time signal processing and sound generation on handheld devices. In *Proceedings of International Computer Music Conference (ICMC)*, 2003.

[5] Gumstix. www.gumstix.com.

[6] S. Schiesser and C. Traube. On making and playing an electronically-augmented saxophone. In *NIME '06: Proceedings of the 2006 Conference on New Interfaces for Musical Expression*, pages 308–313, Paris, France, France, 2006. IRCAM Centre Pompidou.

[7] Z. Settel and C. Lippe. Real-time musical applications using frequency-domain signal processing. In *IEEE ASSP Workshop Proceedings*, 1995.

[8] D. Wessel, M. Wright, and J. Schott. Situated trio: An interactive live performance for a hexaphonic guitarist and two computer musicians with expressive controllers. In *International Conference on New Interfaces for Musical Expression (NIME)*, pages p. 171–173, Dublin, Ireland, 2002.

[9] M. Wozniewski, N. Bouillot, Z. Settel, and J. R. Cooperstock. Large-scale mobile audio environments for collaborative musical interaction. In *International Conference on New Interfaces for Musical Expression*, Genova, Italy, 2008.

[10] A. K. E. Yang and A. T. P. Driessen. Wearable sensors for real-time musical signal processing. In *IEEE Pacific Rim Conference on Communications, Computers and signal Processing PACRIM*, Aug. 2005.