STUDIA UBB MUSICA, LVII, 2, 2012 (p. 41-50)
(RECOMMENDED CITATION)

iFPH: WIRELESS CONTROL OF SOUND[1]

ADRIAN BORZA[2]

SUMMARY. A wireless interactive music system is a powerful tool for creating and performing music on stage. It combines different pieces of hardware and software, and the computer is controlled through a nano control surface or even a smartphone mediated by a nano wireless router. This study focuses on the Max patch for connecting a Wi-Fi device to the iFPH music software.

Keywords: Computer Music, Interactive Music Systems, Max/MSP, Wi-Fi, Wireless, OSC, iFPH, Android, Smartphone, nano Router

1. Premise of Research

The author of this paper has tested recently some limits of the interaction between performer and computer, allowing the computer to act in response to the musician performance, by itself, most of the time. The computer has been involved in performance with its own sound control data. This approach could be considered a disadvantage when it comes to musical improvisation on stage, but the computer is likely an important benefit for an accurate, synchronized performance of an electroacoustic music score.

Anyway, the author has experienced both approaches during the *remote ctrl* Project. He has presented publicly two musical works composed and programmed by himself: *If for Oboe and Interactive Computer* (2011), and *Akedia for Oboe, nanoKontroler and iFPH* (2011).

The *remote ctrl* Project was an outcome of the collaboration between three composers-programmers, Constantin Basica, Adrian Borza and Cătălin Creţu, supported by the Electroacoustic Music and Multimedia Center from Bucharest (CMeM), the National University of Music from Bucharest (UNMB), and the *Gheorghe Dima* Academy of Music from Cluj-Napoca (AMGD). The *remote ctrl* concerts took place on December 18, 2011 at UNMB, and on January 16, 2012 at AMGD.

[1] This research paper is part of the MIDAS 2012-2013 postdoctoral research calendar, under the National University of Music in Bucharest. The research project is funded by MIDAS – the Music Institute for Doctoral Advanced Studies, POSDRU/89/1.5/S/62923, co-financed by the European Social Fund through Sectoral Operational Program for Human Resources Development 2007-2013.

[2] Currently he is a professor at the *Gheorghe Dima* Academy of Music. Address: 25, I.C. Brătianu, Cluj-Napoca, Romania. E-mail: aborza@gmail.com

2. Interactive Music Systems

By definition, "an interactive music composition and performance system is a real-time composing and sound-producing system which employs a synthesizer, a programmable computer, and at least one performance device […]. The system is interactive in that a user can direct aspects of the system's production of music, as he or she hears it being produced, by use of the performance device."[3]

We preserve here the key-components of an interactive computer music system: performance device, programmable computer, and sound-producing device. Nevertheless, the complexity of interactive music system is the result of myriad of distinct interconnected components.

The devices working together can be conceptually bundled into three sequences or stages of the whole process of music production, in which information is transferred from one device to another. "The first is the sensing stage, when data is collected from controllers reading gestural information from the human performers onstage. Second is the processing stage, in which a computer reads and interprets information coming from the sensors and prepares data for the third, or response stage, when the computer and some collection of sound-producing devices share in realizing a musical output."[4]

A chain of diverse electrical and electronic devices interconnected, which is defined in this study as interactive music system, is a powerful tool for creating and performing music on stage nowadays. The system merges several pieces of hardware, such as a microphone, a mixer, a sound card, an audio control surface, a wireless router, and even a Wi-Fi mobile phone, but not exclusively. The interactive system is built around a computer and driven by specialized software, yet it is flexible and portable. Its main abilities are that it analyses the performer's input data and it responds instantly to the musician actions during a live performance[5].

3. Integrating Wireless Control

Our interactive music system is built around a laptop driven by the iFPH music software. The software is coordinated by Performer 1 (see Table 1) through a nano audio control surface, but also it integrates a wireless control mechanism over iFPH. The Performer 1 is the actual instrument performer.

---

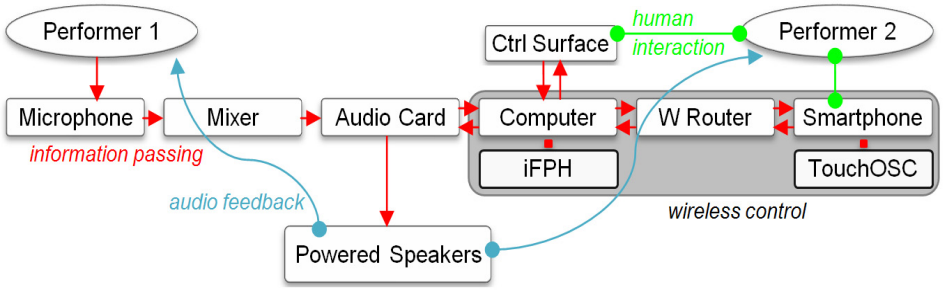
[3] Chadabe, J., *Interactive Composition and Performance System*, United States Patent and Trademark Office, Patent No. 4,526,078, Julie 2, 1985.

[4] Rowe, R., *Interactive Music Systems: Machine Listening and Composing*, The MIT Press, Cambridge, Massachusetts, 1993, p. 10.

[5] For more details about this topic see Borza, A., *Prolegomena to Interactive Music Systems*, Studia UBB Musica, LV, 1, 2010.

Table 1

iFPH Wireless Interactive Music System
Chain of devices and software



Let's have a look at the process of passing information from a device to another:

Sensing stage

- Performer 2 interacts directly with the TouchOSC's user interface on the smartphone's touch screen. The device is capable of reading tactile gestures of performer
- TouchOSC sends OSC messages to iFPH on computer, through the nano wireless router
- The gestural information of Performer 2, collected and converted by the nano audio control surface, is sent to iFPH as MIDI messages
- Information coming from Performer 1 is captured by microphone, pre-amplified by mixer, digitized by sound card, and sent to iFPH

Processing stage

- There is a software module built into iFPH – *Wireless Control* – which is programmed to receive and transmit OSC messages. Roughly, the module has the role to bind the OSC data to the processing data sets. These sets are stored into the iFPH's *Presets* module
- iFPH interprets as well the MIDI data coming from Performer 2 and binds these data to the processing data sets

Response stage

- As an immediate result, iFPH puts into practice its transformational abilities – successive manipulations of the input sound – then sends its output audio signal to the self-powered speakers, back through the sound card

We should mention the next sequence of transmission of information through the iFPH wireless interactive music system:

Feedback stage

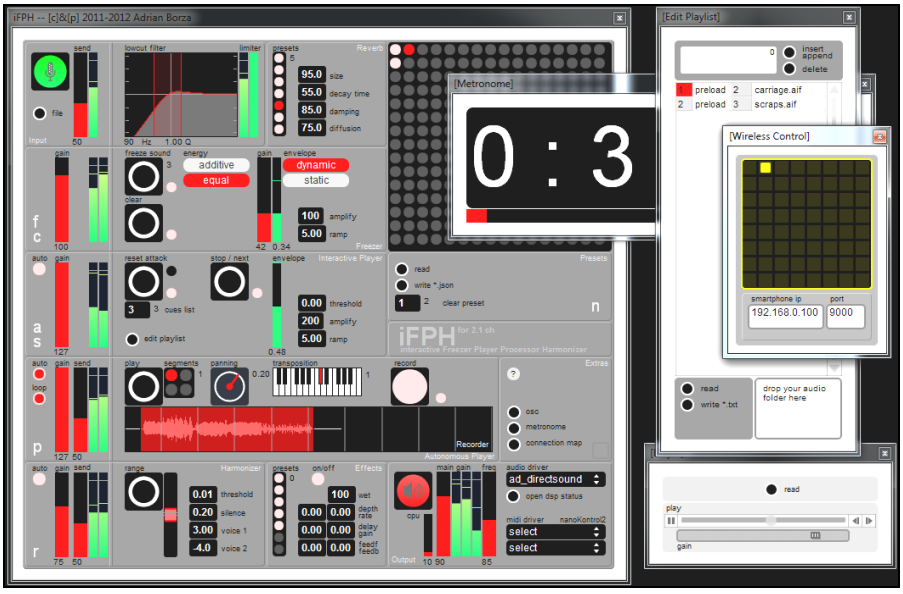
- The *Wireless Control* module sends back OSC messages towards TouchOSC, through the nano router. The messages are exposed visually on the smartphone, thus confirming the Performer 2's gestures
- iFPH sends back MIDI messages towards the nano control surface for the same reason
- Both musicians will find audio feedback of their performance, while it being produced, by listening to music through the loudspeakers

4. iFPH Music Software

What does iFPH stands for? The peculiar name stands for Interactive, Freezer, Player, Processor, and Harmonizer, suggesting some of the actions that it is capable of doing. The iFPH software, assembled with Max/MSP by the author of this study, was constantly developed, therefore the current version 2.5 (2012) is far more advanced than the initial functional version. The features of the current version are briefly described below.

Table 2

iFPH Music Software – the graphic user interface



High-pass Filter: the input audio signal is filtered according to the cutoff frequency, attenuating the amplitude of unwanted low frequencies but passing higher frequencies. The visible range of the filter on the GUI is from 20 Hz to 1 kHz.

Freezer: a sound grain is extracted from the input signal and after that is looped, layered and enriched, creating and sustaining a derivate sound. The overall amplitude of the new sound is static or dynamic, depending on the operation mode selected in the *Envelope Follower for Freezer* module. Up to 8 different sounds can be superimposed. The synthesis method is granular.

Envelope Follower for Freezer: the overall amplitude of the input audio signal is instantly applied to the audio signal generated by the *Freezer* module, personalizing the derivate sound. This means that the module reacts to music dynamics, and the input envelope is monitored constantly.

Interactive Player: pre-recorded sounds or audio files are played back when the overall amplitude of the input signal reaches or exceeds a specified threshold. The files are organized into the *Playlist* module.

Playlist: this is a GUI editor for organizing sound files, built with the purpose to be played back in sequential order, by using the *Interactive Player* module. The file types are AIFF and WAV.

Recorder: the input audio signal is recorded onto a memory buffer, and then is automatically segmented in small fragments. Segmentation is based on the signal's overall amplitude. The maximum recording length is 10 seconds, and the number of audio segments is 4.

Autonomous Player: each time an audio segment is played back, random panning and transposition is applied to it. This is especially true if *Auto* and *Loop* modes are on.

Harmonizer: performs a FFT and pitch-shifting in frequency domain of the input signal, building a chord of 3 sounds. The transposition range is from -24 semitones to +24.

Effects: in *Wet* operation mode, the input audio signal is processed by adding effects, such as distortion, chorus, phaser or flanger.

Reverb: the input signal is processed by adding reverb effect.

Metronome: intended to help musicians at the time they practice and perform on stage, this module is a 60 BPM visual click track synchronized to the first sound file on the *Interactive Player* module.

Crossover: the output audio signal is split into 2 frequencies bands and then routed to the first 3 channels of the sound card, achieving a simple, but productive bass management.

Presets: all the processing data sets (sets of control data over the sound) are saved into a file and called anytime later at musician convenience, on stage. The maximum number of presets is 240. The *Presets* module is controlled by the smartphone or the nano control surface.

Wireless Control: this is an 8x8 grid of presets, a network connection setup, and a module for sending, converting and receiving OSC messages. The module is controlled by the smartphone.

The key-modules *Freezer*, *Envelope Follower for Freezer*, *Interactive Player*, *Recorder*, *Autonomous Player*, *Harmonizer* and *Presets* are controlled by the nano control surface, in addition by using a mouse and a computer keyboard.

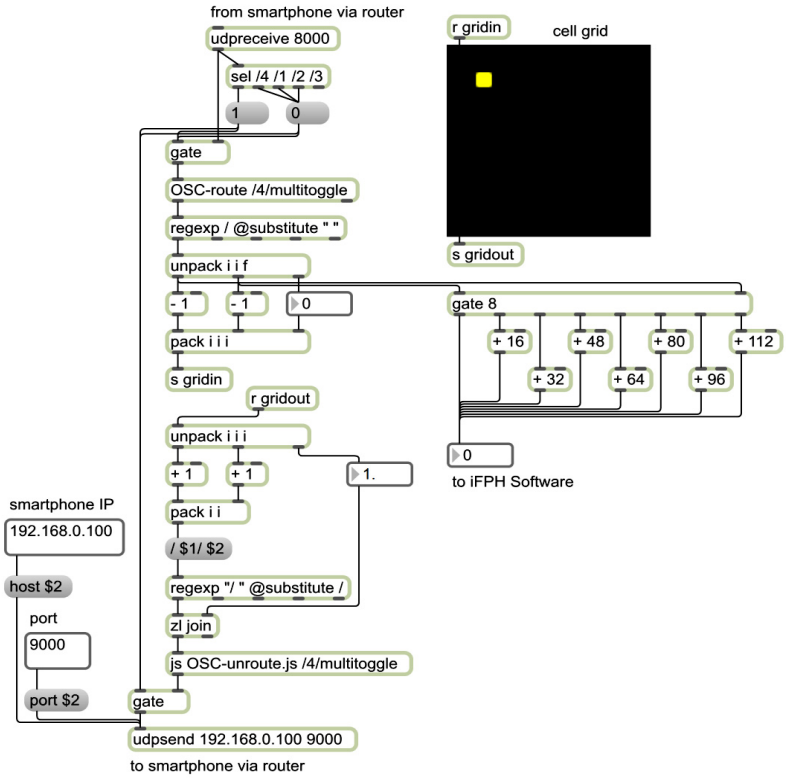
4.1. iFPH Wireless Control Module

The 8x8 grid of processing data sets, called *Wireless Control*, is a Max patch file. The patch has been conceived in 2012 by the author of this research paper, with the purpose of offering more flexibility in using the iFPH software during concert.

We will describe in detail the functionality of the patch.

Table 3

iFPH Music Software – the Wireless Control module



The *udpreceive* object works in conjunction with the Open Sound Control (OSC) protocol[6], and it receives messages transmitted over a wired or wireless network using User Datagram Protocol (UDP). The argument **8000** is the port number for receiving OSC messages from a smartphone via the nano router[7].

[6] Open Sound Control protocol has been developed at the Center for New Music and Audio Technologies (CNMAT) at the University of California in Berkeley.

[7] See below *Smartphone/Computer Wireless Network Connection Setup.*

The *sel* and *gate* objects are working together in order to bypass OSC data only when the page 4 is selected on the TouchOSC's user interface[8]. Otherwise the *gate* object will stop the communication.

The *OSC-route*[9] object matches *multitoggle* element on the page 4, as instructed into its argument, **`/4/multitoggle`**, then it passes any OSC message matched before. The massage represents a cell, defined by column, row, and the on/off state value of the *multitoggle*.

The special *regexp* object substitutes a symbol within the matched OSC message, and then it is divided into individual messages by the *unpack* object.

At this moment, the *gate* object will pass the column number only through the output associated to the specified row number, on one hand. The argument **8** is the number of gates, seen as rows' number. The *+ (addition)* objects bind the column numbers to the preset numbers stored into the *Presets* module. The messages are sent to iFPH.

On the other hand, the *- (subtraction)* and *pack* objects will prepare the messages for the *matrixctrl* object. This object is a cell grid which it resembles the TouchOSC's *multitoggle* interface element. Any action on the *multitoggle* will cause a response on the *matrixctrl*'s interface. In other words, any action of the musician on the smartphone causes a visual feedback on the computer. And vice versa.

The processes are constructed then in reveres: the column, the row, and the state value of the *multitoggle* are joined into an OSC message, so as to be transmitted through the *OSC-unroute.js* and *udpsend* objects. The argument **`192.168.0.100`** is the Internet Protocol (IP) address of the smartphone, and the argument **`8000`** is the port number for sending OSC messages to the smartphone via the wireless nano router. The arguments are changeable.

5. Smartphone/Computer Wireless Network Connection Setup

In order to achieve control over the iFPH with a smartphone, it must first install the mandatory software on the mobile phone and laptop, then to set up a wireless network connection.

There are several applications for Android OS capable to send and receive OSC messages over Wi-Fi network, but we decided to explore in depth two of them, TouchOSC 1.3 and Control 1.31. We have made use of both runtime and fully-functional version 5 and 6 of Max/MSP to run iFPH successfully.

[8] TouchOSC is a virtual audio control surface for iOS and Android operating systems, developed by Hexler.

[9] OSC-route and OSC-unroute.js are Max external objects developed by Mat Wright at CNMAT.

Also, we have effectively tested the following hardware and software configuration of our wireless interactive music system:

Smartphone Applications: TouchOSC 1.3, Control 1.31
Smartphone Operating System: Android 4.0.3
Smartphone Type and Model: Galaxy S II, No GT-I9100
Audio Control Surface Type and Model: Slim-line USB Control Surface, nanoKontrol2
Computer Applications: Max 5.1.9, Max 6.0.5, iFPH
Computer Operating System: Windows 7 64-bit
Computer Model: HP ProBook 4530s
Wireless Router Type and Model: TP-Link 150Mbps Wireless N Nano Router, No TL-WR702N

5.1. Connection Setup Tutorial

Computer Configuration: Windows 7 64-bit

- Connect the router to the computer via USB ports
- Open *Wireless Network Connection Status* of the active wireless adapter by following these steps:
    - Press *Start*, select *Control Panel*, *View network status and tasks* under *Network and Internet*, *Change adapter settings*, then double-click on *Wireless Network Connection* of the active wireless adapter
- Press *Properties* on the *Wireless Network Connection Status* window
- Select *Internet Connection Version 4 (TCP/IPv4)* and press *Properties* on the *Wireless Network Connection Properties* window
- Activate *Use the following IP address* on the *Internet Connection Version 4 (TCP/IPv4) Properties* window, then follow these steps:
    - Write **`192.168.0.200`** in the *IP address* field. This is the IP address of the computer; anything from **`192.168.0.1`** to **`192.168.0.253`**
    - Write **`255.255.255.0`** in the *Subset mask* field
    - Write **`192.168.0.254`** in the *Default Gateway* field. This is the router's IP address
    - Close all windows with *OK* and *Close*. Done

Verify the computer's IP

- Write **`ipconfig`** on the command line of the *Command Promp* (DOS) application and press *Enter* on the computer keyboard
- *Wireless LAN adapter Wireless Network Connection* should provide you with this information:

```
Windows IP Configuration
Wireless LAN adapter Wireless Network Connection:
   IPv4 Address. . . . . . . . . . . : 192.168.0.200
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.0.254
```

Wireless Router Configuration: TL-WR702N

- Write http://192.168.0.254 into any browser's address bar and login with the router's credentials
- Select *Working Mode* under *Basic Settings*
- Select *Router: Wireless router mode* on the *Wireless Working Mode Settings* window, and press *Save*. Done

Smartphone Configuration: TouchOSC 1.3 Connection

- Install TouchOSC 1.3 on the smartphone
- Activate Wi-Fi connection
- Open TouchOSC, and select *OSC* under *Settings* /*Connections*, then follow these steps:
    - Write **`192.168.0.200`** in the *Host* field. This is the IP address of the computer; see *Computer Configuration* for details
    - Write **`8000`** in the *Port (outgoing)* field. This is the port for sending OSC messages to iFPH
    - Write **`9000`** in *Port (incoming)* field. This is the port for receiving OSC messages from iFPH. Done

Verify the smartphone's IP

- Select *Settings*, *Wi-Fi*, and the connected router: *TP-LINK_xxxxxx*
- Verify *IP address* on the mobile phone: **`192.168.0.100`**

6. Smartphone/Computer Software Configuration Tutorial

Smartphone Configuration: TouchOSC 1.3 Layout

- Open TouchOSC, and select *Layout* under *Settings*
- Select *Simple*, press *Done*, and select the last page of the *Simple* Layout. The *multitoggle* graphic interface element is connected automatically to the *Wireless Control* module of iFPH. Done

Max 5/6 and iFPH Configuration

- Install Max 5 or 6 Runtime, then *OSC-route.mxe* and *OSC-unroute.js* external objects into the max-externals folder of Max 5/6
- Install iFPH. Done

7. Conclusion

The non-linear access to the iFPH's processing presets reflects probably the universal approach in live computer composition and performance, that fits a musician, in the sense that Performer 2 would prefer to "jump" or switch anytime to any preset represented visually on the smartphone's touch screen. Spontaneous creation and re-creation of music are natural processes on stage. No doubt, the flexibility in choosing one or other processing data set is an advantage when it comes to improvisation. Improvisation is unpredictable, as we all know. By contrast, the non-linear access is significantly diminished when the very same musician reproduces linearly an electroacoustic score, by using an interactive music system.

Apparently, an interactive music system is a remarkable computer tool for making music, thus the role of the interactive system is that of a musical instrument, and has to be mastered. But what is beyond this role?

"If we are not capable of creating roles for the computer which go beyond our own image, perhaps we can at least attempt a synthesis of all these anthropomorphic and musical roles. This hybridization would at least offer us a glimpse of other functionalities, other conceptual frameworks, and can only serve to enrich the musical discourse and interactivity of real-time computer music."[10]

REFERENCES

Borza, Adrian, *IAC: Interactive Music Systems*, Studia UBB Musica, LIV, 1, 2009, pp. 123-128.

Borza, Adrian, *Prolegomena to Interactive Music Systems*, Studia UBB Musica, LV, 1, 2010, pp. 49-58.

Chadabe, Joel, *Interactive Composition and Performance System*, United States Patent and Trademark Office, Patent No. 4,526,078, Julie 2, 1985.

Lippe, Cort, *A Look at Performer/Machine Interaction Using Real-time Systems,* ICMC, Proceedings, Hong Kong, 1996, pp. 116-117.

Rowe, Robert, *Interactive Music Systems: Machine Listening and Composing*, The MIT Press, Cambridge, Massachusetts, 1993.

[10] Lippe, C., *A Look at Performer / Machine Interaction Using Real-time Systems,* ICMC, Proceedings, Hong Kong, 1996, p. 117.