Synchronization of Multimodal Recordings for Musical Performance Research

Javier Jaimovich
Sonic Arts Research Centre
Queen's University, Belfast
Northern Ireland, BT7 1NN
+44 (0) 28 9097 4829
javier@jaimovich.cl

R. Benjamin Knapp
Sonic Arts Research Centre
Queen's University, Belfast
Northern Ireland, BT7 1NN
+44 (0) 28 9097 4069
b.knapp@qub.ac.uk

ABSTRACT
The past decade has seen an increase of low-cost technology for sensor data acquisition, which has been utilized for the expanding field of research in gesture measurement for music performance. Unfortunately, these devices are still far from being compatible with the audiovisual recording platforms which have been used to record synchronized streams of data. In this paper, we describe a practical solution for simultaneous recording of heterogeneous multimodal signals. The recording system presented uses MIDI Time Code to time-stamp sensor data and to synchronize with standard video and audio recording systems. We also present a set of tools for recording sensor data, as well as a set of analysis tools to evaluate in real-time the sample rate of different signals, and the overall synchronization status of the recording system.

Keywords
Synchronization, Multimodal Signals, Sensor Data Acquisition, Signal Recording.

1. INTRODUCTION
With the on-going expansion of sensor technology and low cost acquisition systems, the number or research projects involving gesture measurement in musical performance has increased significantly in the last decade. Unfortunately, this has also led to an increase in the number of recording standards, file formats and communication protocols. This, besides making difficult the task of exchanging and relating data between researchers [1], produces a particular problem when attempting to simultaneously record multimodal signals from an array of heterogeneous sources.

When designing an experiment, researchers face a problem if they want to use multiple acquisition systems (Arduino, I-CubeX, La Kitchen, etc.), store data in different computers simultaneously, or use equipment with different sampling rates. Starting a recording at the same time in two different machines will not necessarily provide two accurately synchronized recorded data streams. While this can be "hand corrected" when using kinematic, audio or visual signals, the accuracy of the results are quite suspect. This technique is particularly poor when recording signals that cannot be manually compared (e.g. physiological signals).

The audiovisual industry has provided several more accurate solutions for synchronizing audio and video data streams between different platforms and technologies, which must be taken into account when incorporating sensor data acquisition systems. Therefore, the problem we address in this paper is how to effectively synchronize sensor data between different data acquisition devices. And make it compatible with audio and video synchronization protocols, using standard equipment that can be adapted for different recording scenarios.

In summary, we present a system that:

- Synchronizes multiple sensor data signals
- Allows synchronization with standard audio and video software applications
- Works with different technologies and platforms
- Provides a set of testing tools for system evaluation

2. DEVICE, TRANSMISSION AND PLATFORM POTENTIAL PROBLEMS
Table 1 shows only a sample of commercially available acquisition devices for capturing sensor data, obtained from the SensorWiki project[1]. Each different acquisition device has different sampling rate options and communication protocols, which leads to potential synchronization problems at both ends. On one hand, despite manufacturer's indication of sample rates, either via the device's specifications or by a software interface to configure the device, these may vary significantly from the specified value.

Table 1. Sample of acquisition devices listed by SensorWiki

Product	Manufacturer	Sampling rate	Connection / Protocol
Arduino	Arduino	15 ksps @ 10 bit	USB, serial
Wiring i/o board	Wiring	15 ksps @ 10 bit	USB, serial

[1] Full list is available at http://www.sensorwiki.org [accessed 01 February, 2010]

microDig	I-CubeX	1.5 ksps max (1 ch. @ 7 bit)	MIDI
USB-microDig	I-CubeX	5.7 ksps max (1 ch. @ 7 bit)	Serial over USB
Arduino BT	Arduino	15 ksps @ 10 bit	Bluetooth
Make Controller	Making Things	384 ksps @ 10 bit	USB, Ethernet
Wi-microDig	I-CubeX	5.7 ksps max (1 ch. @ 7 bit)	Serial over Bluetooth

Figure 1 shows a plot of the time difference between an Arduino[2] and a Wi-microDig[3] micro-controllers, which were configured to sample data at 250Hz[4]. Both devices were receiving the same input signal; an electrical pulse sent every 5 seconds. The variation, of approximately 100ms per minute, is explained by the difference between the actual sample rate (SR) of the acquisition devices, or acquisition chain (micro-controller + transmission protocol). A test that recorded the number of samples transmitted by the device during 1 minute, and then averaged this data over 10 repetitions, revealed that the actual SR for these particular devices was 250.032Hz ($\sigma = 0.073$) for the Arduino, and 250.497Hz ($\sigma = 0.083$) for the Wi-microDig.

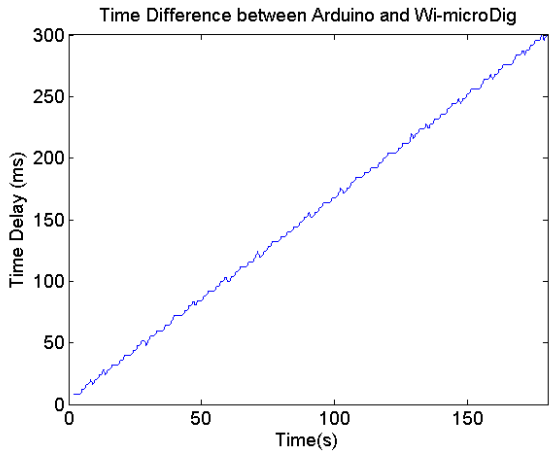

Figure 1. Example of time differences obtained between an Arduino and a Wi-microDig configured to transmit at a SR of 250 [Hz]

Moreover, depending on the communication protocol with the computer (USB, serial, Bluetooth, Ethernet, etc.), the transmission can experience packet loss and jitter problems.

In regard to audiovisual software and hardware platforms, it is beyond the scope of this paper to analyze the differences between technologies and manufactures, but it is important to mention that most available audiovisual platforms have the option to interconnect and synchronize (as master or slave) to external machines. This is made possible by the implementation of time code (TC) protocols developed by the industry, such as SMPTE/EBU, LTC, VITC, and MTC among others [2].

Nevertheless, the fact that a multi-track audio recording platform is capable to 'slave' its clock to an external source, does not necessarily imply that an audio track recorded in that session will be precisely synchronized with the clock. Three different multi-track recording platforms tested by the authors had different latency times with respect to the clock; Pro-Tools, Sony Vegas and Adobe Audition. These variations are produced by the different sizes of the internal buffers used for recording, and do not signify an error in the design of the platform. Even though in most cases the buffer size can be modified depending on the hardware capacities of the host computer, it cannot be completely removed and should be a factor when addressing synchronization issues.

In summary, a time variation between recorded signals can be originated in multiple parts of the transmission path. It could be due to SR differences, latency or packet loss in the communication protocol, or different processing times in the recording platform.

3. RECORDING SYSTEM

Regardless of the origin of the possible time variations described in the previous section, it is of utter importance to be able to monitor and test the time of arrival for each individual signal recorded. For this, a 'master' time code (TC) signal must be sent to every recording device. This allows a real-time analysis of each incoming signal's sampling rate, and can later be used to analyze and compare the differences between each section of the recording system.

3.1 MIDI Time Code

MIDI Time Code (MTC) is a standard developed for transmitting SMPTE-based time code through the MIDI protocol. MTC uses the quarter-frame MIDI messages to embed the SMPTE time code, and it is widely incorporated in many operating systems to lock MIDI sequencers, digital audio workstations, etc., and also to synchronize with video devices by the use of a SMPTE-to-MTC converters [3].

Due to the ubiquity and simplicity of MIDI interfaces, which allow several computers to receive TC by just sharing a MIDI connection, a recording system was developed using MTC to timestamp every sample recorded, and to sync the audio and video recording systems.

The recording system requires a master clock, which is sent to all the recording devices, and can be generated by either an audio or video device with SMPTE (with a SMPTE-to-MTC converter), or a standalone clock interface such as the MOTU Midi TimePiece[5].

3.2 Max/MSP tools

A set of tools was developed for Max/MSP[6] to record sensor data and measure sampling rate in real-time. The recording patch receives and decodes MTC from a MIDI interface, and attaches a time-stamp to every sample recorded (see Figure 2). The SR tester object, calculates the SR of any incoming signal by filling a small buffer with the data stream and calculating the number of samples recorded against the recorded time.

3.3 System Configuration

Figure 3 shows an example recording configuration diagram used for an experiment carried out by the authors. The signals recorded in the session included video from two cameras, four channels of audio, kinematic and physiological signals from

[2] www.arduino.cc [accessed 01 February, 2010]

[3] www.infusionsystems.com [accessed 01 February, 2010]

[4] A firmware using interrupt handlers was written to obtain a periodical SR in the Arduino.

[5] www.motu.com/products/midi/mtpav_usb/ [accessed 01 February, 2010]

[6] www.cycling74.com [accessed 01 February, 2010]

four musicians (8 channels each), and physiological signals of 9 audience members (2 channels each). For this session, four different computers were used to record video, audio and sensor data. Moreover, two computers were used to receive signals from wireless acquisition devices and send them through OSC to the data-recording computer.

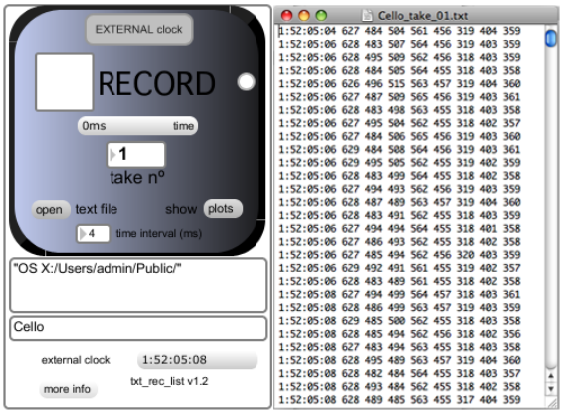

Figure 2. Sensor data recording object for Max/MSP (left) and example of recorded txt file with TC (right)

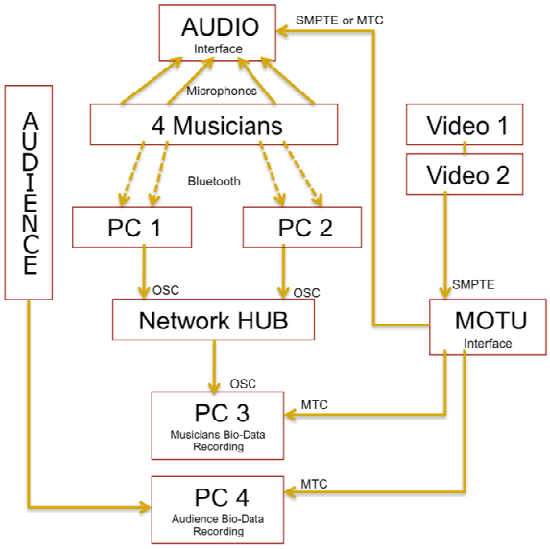

Figure 3. Example configuration diagram of the recording system during an experiment session.

For this experiment session, one of the video cameras generated SMPTE time code, which was sent to a MOTU SMPTE-to-MTC converter that distributed MTC to all recording devices. The audio multi-track recording platform was configured to use MTC as a master clock, and the video from both linked cameras was recorded with the original SMPTE time code.

In another recording session, a different configuration was utilized. The multi-track audio session generated MTC, which was connected to the MOTU interface, and then distributed to all the computers recording sensor data, as well as creating a SMPTE copy to be recorded as an audio track in the tape of the video camera used in this session. The key element for this and any configuration set-up is for the recording devices to receive a copy of the MTC signal (or a converted equivalent, e.g. SMTPE).

3.4 Testing and Calibration of the System

Because the equipment and conditions of a recording session can change considerably between experiments, the main objective of this recording system is to evaluate and register the time differences of the different components. This approach does not focus on attempting to force synchronization between the different devices before recording, but to effectively assess the multiple differences they possess so they can be subsequently corrected in the analysis stage.

To achieve this, a simple signal generator has been implemented using an Arduino and set of simple electronic components, which synchronously outputs a digital, audio and visual pulse train. Before the experimental session, the test signal is sent to all the recording devices, which are also receiving the MTC. Subsequently, every test signal recorded is loaded into a MATLAB algorithm that calculates the differences between the individual devices. These differences are registered and after the recording session, each recorded signal is calibrated and corrected according to its individual temporal displacement.

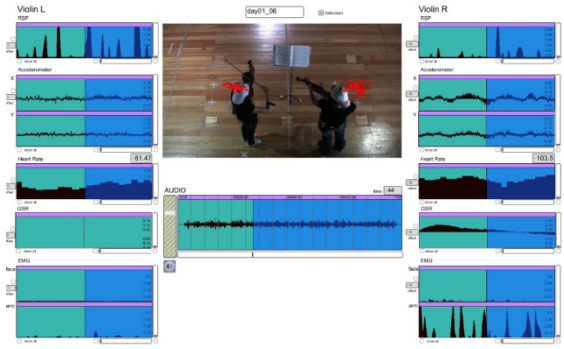

Figure 4. Example of sensor data and audiovisual signals synchronized using the presented system

4. CONCLUSIONS

We have presented a practical system that synchronously records sensor data from different acquisition systems and is compatible with standard audiovisual recording platforms.

The main advantages of the proposed system rely on its modular capacity, being able to adapt to different recording scenarios. Furthermore, the system corrects the relative times of arrival of each signal, regardless of the sum of latencies that recording from several dissimilar devices can cause.

Even though we have used this system for several experiment sessions involving multimodal signals, we are still developing the tools and algorithms to make each step as automated as possible. Additionally, work on creating similar blocks to allow EyesWeb [4] integration is under development.

5. REFERENCES

[1] A.R. Jensenius, N. Castagné, A. Camurri, E. Maestre, J. Malloch, and D. Mcgilvray, "A Summary of Formats for Streaming and Storing Music-Related Movement and Gesture data," *Proceedings of ENACTIVE/07*, Grenoble, France: 2007.

[2] T. Amyes, *Audio post-production in video and film*, Focal Press, 1998.

[3] D.M. Huber and R.E. Runstein, *Modern recording techniques*, Focal Press, 2001.

[4] A. Camurri, S. Hashimoto, M. Ricchetti, A. Ricci, K. Suzuki, R. Trocca, and G. Volpe, "EyesWeb: Toward Gesture and Affect Recognition in Interactive Dance and Music Systems," *Comput. Music J.*, vol. 24, 2000, pp. 57-69.