# NUANCE: A SOFTWARE TOOL FOR CAPTURING SYNCHRONOUS DATA STREAMS FROM MULTIMODAL MUSICAL SYSTEMS

*Jordan Hochenbaum*[1,2]
New Zealand School of Music[1]
PO Box 2332
Wellington 6140, New Zealand
*hochenjord@myvuw.ac.nz*

*Ajay Kapur*[1,2]
California Institute of the Arts[2]
24700 McBean Parkway
Valencia CA, 91355
*akapur@calarts.edu*

## ABSTRACT

In this paper we describe Nuance, a software application for recording synchronous data streams from modern musical systems that involve audio and gesture signals. Nuance currently supports recording data from a number of input sources including real-time audio, and any sensor system, musical interface, or instrument which outputs serial, Open Sound Control (OSC), or MIDI. Nuance is unique in that it is a highly customizable to the user and unknown musical systems for music information retrieval (MIR), allowing virtually any multimodal input to be recorded with minimal effort. Targeted toward musicians working with MIR researchers, Nuance considerably minimizes the set-up and running times of MIR data acquisition scenarios. Nuance attempts to eliminate most of the software programming required to gather data from custom multimodal systems, and provides an easy drag-and-drop user interface for setting up, configuring, and recording synchronous multimodal data streams.

## 1. INTRODUCTION

Imagine a common scenario where a researcher is investigating some music related problem. Whether the task is a classification problem, clustering, pattern matching, query/retrieval, musical perception and cognition problem, etc, all tasks share the initial step of acquiring and preparing the data set. While this point seems quite trivial, consider the following. Say the task is a performance metrics problem and the dataset is a collection of features extracted from microphone recordings of a drummer. The researcher would like to perform a similar experiment with a saxophonist. No problem, there are tools the experimenter could easily use to record the audio, perform feature extraction, and finally analysis. This scenario, however, becomes much more difficult when the experiment involves custom instruments, interfaces, and multimodal/multisensory input systems. Let's say the drummer mentioned is playing a drum modified with various sensors on the drumhead and stick, the data of which is to be captured alongside the audio recording. Similarly, an accelerometer and air-pressure sensor measures the characteristics of the saxophone performance. Given the highly individualized nature of working with different instruments and musical contexts, each problem requires a different software tool to be written for acquiring the data set. Imagine being a recording or live sound engineer and requiring a specific piece of hardware, or software plug-in, to interface with each instrument being used in a performance. In this paper, we describe a software tool we have created called Nuance, which begins to address such scenarios. We hope Nuance brings the task of gathering multimodal data sets for MIR one step closer to the ease, usability, and productive workflow refined in traditional Digital Audio Workstations [4].

The remainder of this paper is as organized as follows. Section 2 describes the motivations behind Nuance, based on the shortcomings of other available solutions. Section 3 describes the software architecture and capabilities of Nuance. Sections 4 and 5 detail recent and possible future research (respectively) supported by the software, and lastly conclusions are discussed in section 6.

## 2. BACKGROUND AND MOTIVATION

Before creating Nuance, a number of available software options were considered. While not comprehensive, the tools discussed in this section were the most ubiquitous tools that appeared to fit the required use cases. The main requirement was to output synchronized recordings from a variety of input sources including audio, MIDI, OSC, serial sensor interfaces, and hyperinstruments. **Figure 1** offers an input requirement comparison between five of the available software / framework candidates we studied.

The three candidates represented by fully dashed rectangles in **Figure 1** (Marsyas, Chuck, and the CREATE Signal Library or CSL) are popular programming languages or frameworks that are capable of multimodal data collection. Both Marsyas [12] and Chuck [13], for example, have many features for performing data capturing, analysis, machine learning, retrieval and synthesis. While they are capable of receiving audio, midi, and OSC input streams, they do not currently support general purpose COM/Serial IO. Serial communication is a significant factor as many of the custom interfaces and sensor systems used in these types of scenarios output serial messages. Another key factor in deciding not to use these three candidates was that a major requirement was to use a tool that required little to no programming to operate. With all three candidates, a custom application would have to be written for each particular experimental setup, as well as implementing a synchronization scheme from the ground up. We desired an application that practicing musicians could run independently, and which requires as little technical know-how and investment of time as possible. To do so, the application would need to provide an easily navigable user interface (GUI).

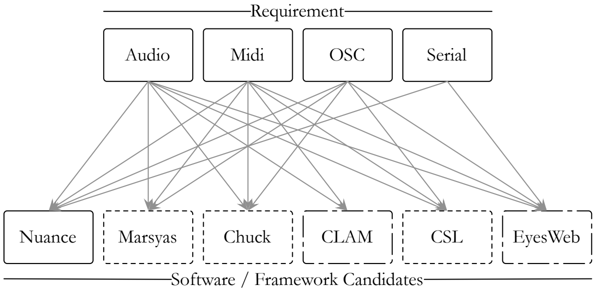


**Figure 1.** Requirement Comparison of other software and frameworks considered

CLAM [1] and EyesWeb XMI [3] (semi-dashed rectangles in **Figure 1**) are two frameworks that offer a wide array of features, in an interactive, node-based patching environment. CLAM includes a Data2Audio transformation module as well as a module to export the audio stream to disk. EyesWeb supports all required input streams, as well as providing support for additional input streams, like motion capture data. Additionally EyesWeb XMI provides a configurable data synchronization scheme. While both options seemed viable at first, they did not meat the requirements in the following ways. Firstly CLAM does not support (to our knowledge) OSC or serial input. Secondly we found that the node-based patching environments of both CLAM and EyesWeb were powerful solutions for configuring many complex scenarios and experimental systems. However, the goal was to utilize a tool that focused solely on data capturing, which unsupervised, could be easily configured and used by practicing musicians. In our trials, having to patch and synchronize each instrumental setup individually was found to be too labor intensive (for our purposes) and a more tailored solution was desired. Other common visual-based programming languages such as Max/MSP and Pure Data are also capable of the desired tasks, and provide additional support or accessing data from other inputs streams, but also shared similar traits.

While the above tools all share the ability to capture data from various sources, and some even provide additional MIR capabilities under one package, none fulfilled our requirements in dealing with the scenarios as described in section 1.

The necessity for a highly adaptable multimodal data acquisition system is growing as analysis tools continue to get better, and as multimodal strategies become more reliable in solving MIR related problems [2, 6, 7]. As previously stated, current solutions require the time-consuming task of writing individualized programs or patches for each instrument or system. This is counter productive as hyperinstruments continue to gain popularity, and as industry continues to produce hybrid digital instruments[1]. In this way, we aim Nuance towards the ultimate goal of being a software solution that enables tapping into these types of instruments, with the ease and usability achieved in the common audio-recording software paradigm. We imagine that it is possible to work within an environment where capturing multimodal musical data, whether during the sessions of an album recording, or for MIR related research, is as easy as working within typical multi-track audio recording software.

## 3. ARCHITECTURE AND IMPLEMENTATION

Nuance has been designed such that it can synchronize and record data from a variety on inputs and modalities. In this section, we provide an overview of the Nuance recording system and its capabilities.

### 3.1. Design Overview

As mentioned in section 2, the primary aim of Nuance was to develop a recording application with a traditional DAW-like workflow. The software should be intuitive for regular musicians, while providing, a high degree of flexibility and support for a variety of heterogeneous input data streams. As such, the following list provides an overview of the main software requirements we set out to fulfill.

- Support for a variety of input sources including audio, MIDI, OSC, and serial sensor interfaces
- Minimal programming required (little to no programming or "patching")
- The ability to save, load, and modify recording setups and sessions
- Easily configurable user-interface

[1] E.g. Gibson HD.6X digital Les Paul Guitar, YouRock MIDI Electric Guitar, Rock Band 3 Stratocaster Pro, Fretlight Guitar

- Recording all data in .wav format for analysis

### 3.2. System Overview

The general flow of the software system is detailed in Figure 2. A user provides various multimodal input streams, which are recorded as audio files. By default, all streams are recorded as 16-bit uncompressed .wav files, at a sample-rate of 44.1kHz. This can be adjusted in the program preferences panel, depending on the requirements and capabilities of the users system, up to 24-bit resolution, and a 192kHz sample-rate.

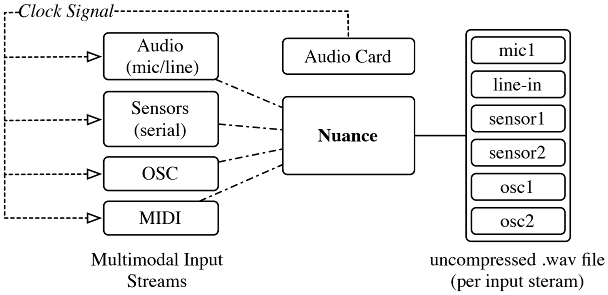


Figure 2. Overview of Nuance

#### *3.2.1. Synchronization*

Nuance implements a synchronization scheme driven by the computer audio cards sample-rate clock (Figure 2). Each sensor or input is responsible for updating itself asynchronously at its own independent rate, and all data-streams are read and recorded within a guaranteed synchronous and thread-safe audio callback system. Whenever a new audio buffer[2] is available, each recorder is simultaneously notified to record its data. For an audio input, this simply means writing its current block of audio. For serial, OSC, and MIDI data, the most recent sample is copied into an array (of equal size as the audio-block) and synchronously written to disk. This sample-and-hold and up-sampling of sensor data happens at a much faster rate than common sensor systems supply new data, and we have found it to be more than sufficient in terms of speed and resolution for MIR applications. Other synchronization schemes are possible, and may be required in the future if additional data sources are added. Additionally, Nuance has been written to support additional output formats (e.g. SDIF/GDIF[3]) in the future.

### 3.3. Multimodal Input

A primary concern with Nuance was to support heterogeneous input channels. While the initial four supported input channels are audio, serial, OSC, and MIDI, the Nuance codebase has been written with future extensions in mind. In the following section we describe Nuances multimodal capabilities in greater detail.

#### *3.3.1. Audio*

Mono audio recording is achieved in Nuance by adding an Audio Recorder track to a Nuance session. Each Audio Recorder has the following parameters: real-time waveform visualization, input channel selector, a gain slider, and a record arm button.

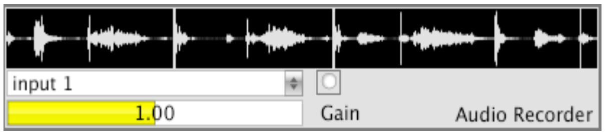


Figure 3. Audio Recorder Object

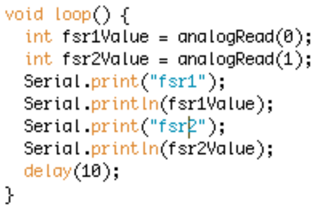
#### *3.3.2. Sensors (Serial Data)*

As many projects in the community utilize Atmel/Arduino/PIC microprocessors, supporting serial communication was a major design consideration. For generalization purposes, Nuance currently supports serial devices outputting data in the following serial format:

| SensorStartMessage | data (10-bit) |
|---|---|

Figure 4. Serial Message Format

A typical use-case using an Arduino microcontroller with two force-sensing resistors connected to analog inputs 0 and 1 might look something like **Figure 5**.



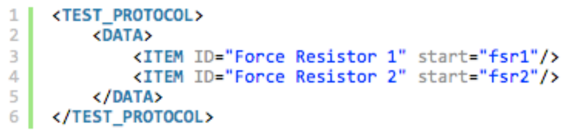
```
void loop() {
  int fsr1Value = analogRead(0);
  int fsr2Value = analogRead(1);
  Serial.print("fsr1");
  Serial.println(fsr1Value);
  Serial.print("fsr2");
  Serial.println(fsr2Value);
  delay(10);
}
```

**Figure 5**. Example Arduino Serial out messages for two Analog Sensors

In this example, "fsr1" and "fsr2" would be the *SensorStartMessage's*, which are immediately appended by the data, and finally followed by a new line character (via println). Nuance uses the new line character to delineate each serial message. Once the serial messages are streaming in the correct format, the user must provide an .xml file (Figure 6) to each sensor recorder object. The .xml file outlines the expected sensor start messages ("start"), paired with a human-readable name ("ID") to show in the Sensor Recorders input selector. A built in message configuration panel is

[2] Buffer-size is adjustable via the "preferences panel"
[3] Sound and Gesture Description Interchange Formats

being considered for a future release, enabling start messages (and paired human-readable names) to be defined and automatically available to all sensor recorders without having to load an .xml file. The serial-protocol currently implemented was designed for simplicity; however, other more optimized protocols are being considered in the future. For serial-based interfaces that cannot conform to the supported protocol format however, it is still possible to capture data via the OSC and MIDI recorder objects.



```
<TEST_PROTOCOL>
    <DATA>
        <ITEM ID="Force Resistor 1" start="fsr1"/>
        <ITEM ID="Force Resistor 2" start="fsr2"/>
    </DATA>
</TEST_PROTOCOL>
```

Figure 6. Example .xml configuration

Each Sensor Recorder has the following parameters: XML-Protocol loading button, record arm button, serial-device selector (which connected serial-device to acquire data from), input range for automatically normalizing incoming data, and a real-time slider to visualize incoming sensor data.

### 3.3.3. *Open Sound Control*

Open Sound Control (OSC) is a versatile communication channel that allows data to be streamed via external sources. The OSC Recorder greatly extends the capabilities of Nuance, making it possible to record data streaming from external applications on the host machine, and from applications and sensor systems connected to networked or remote computers. Additionally, the OSC recorder provides the ability to record sensor-systems or hyperinstruments that do not or cannot follow the generic serial protocol (via a serial-to-osc middleware).

Example external sources can be anything from iPhones and mobile devices, vision tracking and analysis systems, real-time feature extractors, and other derived-data outputs. OSC support allows Nuance to support nearly any input modality and or source natively, while keeping its feature set focused solely on the task of providing high-quality, intuitive multimodal recording.

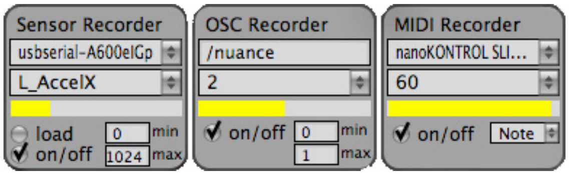


**Figure 7**. Serial, OSC, and MIDI Recorders

### 3.3.4. *MIDI*

The MIDI Recorder enables data from any native MIDI device to be captured in Nuance. Each MIDI recorder can be configured to listen to individual MIDI note or control change (CC) messages from specific devices, including midi-over-network and IAC Bus (Interapplication Control Bus) connections. As all data in Nuance is treated as a continuous stream, when recording MIDI note messages, Nuance does not differentiate between note-on and note-off messages. During analysis however, the rising and falling edges where values transition between zero and the value can be treated as the note-on and note-off instant. In the future, when Nuance supports additional output schemes (such as SDIF/GDIF), note-on and note-off events will be preserved in their normal form.

## 3.4. Workflow

A typical use scenario begins with a new empty *session.* Sessions can be thought of as project files, or serializable experiment configurations. **Figure 8** shows an example Nuance session including several multimodal data-streams.

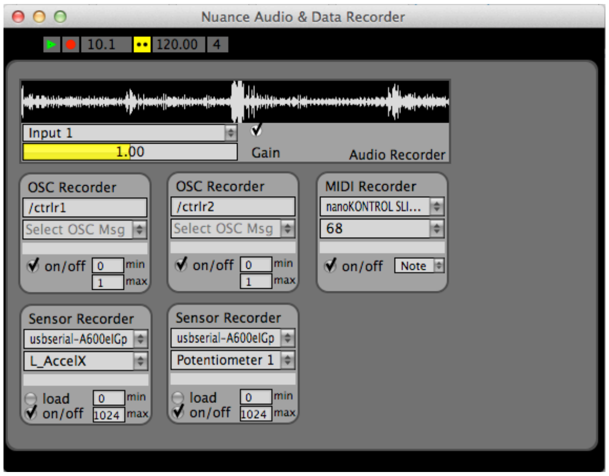


**Figure 8**. Nuance Session Editor Panel

Most of the user-interaction happens in the session editor panel. Right clicking anywhere in the panel brings up a contextual menu that enables various functions to be performed. These functions include adding recorder objects, unlocking the editor panel to resize and position recorder objects, and saving/reloading sessions. Once a session has been configured, it can be saved for reuse at a later time. Modifications to the session can be made any time during the process and re-saved for future use. A built-in metronome and count-in can also be enabled from the main transport, for experimental setups with tightly controlled timing requirements. Lastly, a bar/beat counter is provided to give feedback to the user about how long they have been recording.

# 4. RECENT APPLICATIONS

Over the course of Nuances development, the software has been used in a number of real-world research projects. In this section we present a few examples of uses of the software.

## 4.1. Performer Recognition

One of the first research projects in which an early version of Nuance was employed was for automatic North Indian sitar performer recognition. Nuance was used to record a multimodal dataset including information from thumb pressure, tilt, and fret sensors, as well as the acoustic output of the instrument. The sitar players (spanning beginner to intermediate levels) played a number of traditional North Indian practice exercises, as well as free improvisations to investigate how well the computer could automatically detect a performer based on characteristics of their performance. More information on the experiment and results can be found in [6].

## 4.2. Drum Stroke Computing: Left/Right Hand Recognition & Performance Metrics

The most recent use of Nuance has been for automatic drum-stroke identification and drum performance metrics [7]. Nuance recorded the acoustic output of ten performers playing common snare drum exercises, as well as gestural information from two triple axis accelerometers placed on the hands of each performer. In this experiment, the gestural data from the accelerometers were used only in the training phase of machine learning, enabling the computer to recognize left/right hand drum strokes from the mono audio signal alone. Additionally, this research demonstrated new metrics that can be derived from multimodal analysis of drum performance. Other proposed uses of left/right hand recognition include more comprehensive automatic transcription, computer-assisted musical pedagogy (e.g. interactive rudiment testing and identification), and various uses for control in live-performance contexts.

## 4.3. Long-term Metrics Tracking

Nuance has been used over the last year for long-term performance metrics tracking. Two musicians (one playing a bowed upright bass and the other a custom zither-like stringed instrument) have been using Nuance on their own as part of their weekly practice routines. Both musicians have been practicing with modified instruments providing a variety of data streams which include: tilt and gesture of their bow movements, hand-pressure on the bow, bridge pressure on the instrument, and the acoustic output of the instruments. Active research is underway investigating metrics from the player's performances, such as improvement or trending of rhythm and accuracy over time, improvisatory patterns, and physical and gesture features.

# 5. FUTURE APPLICATIONS

While Nuance is already flexible, and is being used in a number of scenarios, we are extremely excited at its nascent potential. There are many areas where Nuance can mature, and in this section we describe some possible future applications in which we hope to see Nuance used.

## 5.1. Musical Pedagogy

One of our main motivations for Nuance was to create a tool that could lead to what we see as the future of musical pedagogy and performance metrics tracking. We imagine an *Adaptive Practice Room*, one that is acute to a musicians practice, and offers insightful feedback to help guide and inform training musicians. Already some music education programs utilize various software tools assist in musical training. We have even seen the idea proposed of anthropomorphic robotic music instructors that are capable of responding to human performers [10]. Recently, Percival presented an interesting approach to computer-assisted violin practice, and a good overview of the current state of computer-aided musical learning in [9]. While other researchers in the field have shown clear interest, current tools only tend to analyze audio, and only address specific instruments.

Concurrent work in both the MIR community and industry has proven that robust pitch tracking, beat-detection, and other sort of metrics are obtainable. However, there is no solution currently that offers the potential for any practicing musician to play into a system, and get immediate feedback about his/her performance. Nothing yet provides musicians, in any musical context, with the ability to easily track their performance metrics over time. While tools exist for MIR researchers and computer scientists to perform individual experiments, no tool exist which is immediately accessible to practicing musicians, not to mention in the multimodal domain. Although Nuance currently focuses on the acquisition of multimodal data, we hope to support analysis directly in Nuance (or a sister application) in the future. Exporting statistical metrics in a common spreadsheet format would also be a useful first step in this direction. In this way, we hope for Nuance to help bridge this gap between the work happening in the MIR field, the musical classroom, and the way in which future musicians will practice.

### 5.2. Computational Musicology

We also hope to use Nuance in the future for musicological research. We feel tools like Nuance will prove to be elemental in the preservation of multimodal digital fingerprints of musicians across many musical contexts for future generations. We hope to one day have a digital library not only of musical audio recordings and scores, but also the transcribed performance subtleties of great performers of the past.

### 5.3. In The Digital Audio Workstation

Lastly we look forward to a future where multimodal data streams can be integrated into the general workflow of recording and composing as an electronic musician. Working with multimodal musical instruments affords many unique artistic possibilities, from directly manipulating sound parameters, to extracting higher-level features and mapping them to control parameters. Limited ‘general’ tools exist which begin to facilitate these interactions outside of the research laboratory [5], and we hope for Nuance to help guide the way in making this accessible to today’s electronic musicians and composers. With this in mind, we have written the core of Nuance such that it would remain unchanged in the future if we were to author a cross-platform version in VST/Audio Unit/RTAS plug-in formats.

## 6. CONCLUSION

In this paper we have described Nuance, a software tool for recording synchronous multimodal data streams. Nuance currently supports high-resolution audio input, as well as input from nearly any musical instrument or sensor system via serial, OSC, and MIDI protocols. Nuance is different from other solutions in that it is a no patching, near-codeless solution. Nuance was designed to be operated by musicians and researchers alike, and has already been used in many real-world scenarios including performer recognition, drum-stroke identification, and performance metrics tracking. Not only can Nuance increase productivity in MIR scenarios, but we hope it points to and establishes a foundation for other future musical endeavors, both in the MIR-laboratory, the studio, and the musical classroom.

## 7. REFERENCES

[1] Amatriain, X. et al. 2006. CLAM: a framework for efficient and rapid development of cross-platform audio applications. *Proceedings of the 14th annual ACM international conference on Multimedia* (New York, NY, USA, 2006), 951–954.

[2] Benning, M. et al. 2007. Multimodal Sensor Analysis of Sitar Performance: Where is the Beat? *Proceesings of the IEEE International Workshop on Multimedia Signal Processing* (Crete, Greece, 2007).

[3] Camurri, A. et al. 2007. Developing multimodal interactive systems with EyesWeb XMI. *Proceedings of the 7th international conference on New interfaces for musical expression* (New York, NY, USA, 2007), 305–308.

[4] Duignan, M. et al. 2010. Abstraction and activity in computer-mediated music production. *Computer Music Journal (CMJ)*. 34, 4 (2010), 22–33.

[5] Fiebrink, R. 2011. *Real-time Human Interaction with Supervised Learning Algorithms for Music Composition and Performance*. Princeton University.

[6] Hochenbaum, J. et al. 2010. Multimodal Musician Recognition. *Proceedings of the International Conference on New Interfaces for Musical Expression* (Sydney, Australia, Jun. 2010).

[7] Hochenbaum, J. and Kapur, A. 2012. Drum Stroke Computing: Multimodal Signal Processing for Drum Stroke Identification and Performance Metrics. *International Conference on New Interfaces for Musical Expression* (2012).

[8] Kapur, A. et al. 2007. Pedagogical Transcription For Multimodal Sitar Performance. *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)* (2007).

[9] Percival, G. and Schloss, A. 2008. *Computer-Assisted Musical Instrument Tutoring with Targeted Exercises*. University of Victoria, Masters Thesis Interdisciplinary Studies (Computer Science and Music).

[10] Petersen, K. et al. 2008. Development of the Waseda Flutist Robot No. 4 Refined IV: Implementation of a real-time interaction system with human partners. *IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)* (Scottsdale, AZ, 2008).

[12] Tzanetakis, G. and Cook, P. 1999. MARSYAS: a framework for audio analysis. *Organized Sound*. 4, 3 (1999), 169–175.

[13] Wang, G. 2008. *The ChucK Audio Programming Language: A Strongly-timed and On-the-fly Environ/mentality*. Princeton University,.