# MULTI-DIMENSIONAL CONTROLLERS, EXPRESSIVITY, SOFTWARE: AN INTEGRATION PROBLEM

*Jean-Baptiste Thiebaut*
ROLI
jean-baptiste@weareroli.com

*Sean Soraghan*
ROLI
sean@weareroli.com

*Lauren Ianni*
ROLI
ljianni@post.harvard.edu

## RÉSUMÉ

This paper identifies a problem faced by many manufacturers of multi-dimensional controllers (MDCs) for musical control. Namely, the difficulty faced by these manufacturers when trying to integrate their hardware with 3rd party audio software. It presents this problem through the explanation of a new MDC - the Seaboard - and the difficulty being faced with the integration of the Seaboard into audio software environments. A list of parameters is put forward that may facilitate the categorisation of gestures in new MDC systems. These parameters are suggested in order to aid in the development of a new MDC framework that focuses on gestural control and relevant musical output. A solution to this specific protocol is beyond the scope of this paper ; we're setting the context for gathering academic and industrial feedback to design an appropriate protocol for MDC communication and integration.

## 1. INTRODUCTION

Multi-Dimensional Controllers (MDCs) have the potential to facilitate creative expression in music as well as in other fields such as music therapy and social interaction. We distinguish discrete controllers - whose mapping requirements are addressed by MIDI - from continuous controllers, whose richer, higher definition data, is inadequately handled in existing musical data protocols. This paper does not propose a specific protocol, rather it presents a higher level framework for the categorisation of new sets of gestures, that could be implemented in existing protocols to enhance musical creation and facilitate software compatibility and lower the barriers of adoption.

There has been an increase in the development of MDCs, yet the method of encoding abstract musical data in computers remains grounded in keyboard-based discrete controllers. Indeed, musical expression that uses controllers is currently limited by the boundaries of the keyboard paradigm, where the note begins when the movement finishes and focuses mainly on discrete pitches and durations. In the field of music technology, professionals push the boundaries of their design, but their output is compromised by the limitations of discrete and low resolution encoding protocols, such as MIDI. On the other hand, high definition protocols, such as OSC, are too open to allow versatile and easy integration in commercial sound software. While it is becoming increasingly easy to get access to sophisticated sound design tools, it is significantly harder for musicians to personalise and be creative with those sounds, due to the limited interaction available. In order to increase the playability of sound software, controllers require a change in form factor and new ways to encode data.

## 2. THE SEABOARD

The Seaboard is a new tangible keyboard instrument, which facilitates intuitive music creation [6]. The intent of the product design is to deliver an integrated music-creation device, which merges traditional keyboard design with modular technology. The Seaboard is the first application of the patent-pending SEA (Sensory, Elastic, Adaptive) technology.

**Figure 1**. The Seaboard

The Seaboard is an MDC which enables both continuous and discrete control through note-by-note real-time continuous polyphonic control of pitch, amplitude, and timbral variation. In contrast to previous, limited keyboard and sound production interfaces, the Seaboard diversifies and maximizes the degree and types of musical expressivity through its ability to simultaneously control continuous and discrete aspects of sound. The Seaboard is a continuous action interface (CAI), a system which registers spatial or gestural movement in time to enable more complex inputs based on continuous movement. The Seaboard‘s continuous functions include glissando/slide effects, timbral, and dynamic variations in real time. Its continuous control includes the ability to gather and map rich sets of data in a variety of ways. The Seaboard is also a discrete control interface (DCI), a system with inputs which can be distinguished in time. The Seaboard‘s discrete functions include inputs to generate the notes of the chromatic scale. The Seaboard‘s discrete controls include analog (usually switch-based) controls that simulate a mechanical action.

### 2.1. Integration Challenges

One of the most significant challenges during the development of the Seaboard has been the translation of the rich output data produced by the Seaboard into a structure that third party audio software programmes can interpret. The majority of existing hardware controllers and software programmes make use of the MIDI protocol. Attempts to integrate the Seaboard with the MIDI protocol have helped to highlight some specific issues that exist with MIDI.

When polyphonic synthesizers were first being developed during the late 1970s, one major issue was the inability for different manufacturers‘ machines to communicate. MIDI was introduced as a solution to this problem. Although the introduction of MIDI was a major breakthrough in computer music, issues arise from the fact that it is based on the physical keyboard interface.

#### *2.1.1. Limited Note Control*

On a traditional keyboard interface, a note begins when the movement finishes, when the hammer hits the strings, and ends when the key is released. The users only have control over the beginning and end of a note. They have no control over the parameters of that note over the duration that it plays. The MIDI protocol extends this notion. In MIDI, there are ‘note on’ and ‘note off’ values. The volume, pitch and timbre of these notes are determined by the channel in which the notes originate.

#### *2.1.2. Pitch Bending*

When synthesisers were first developed, the idea of pitch-bend manifested as a controllable wheel, that globally altered the pitch output. Altering the position of this wheel would change the pitch of every note that was played on the synthesiser. MIDI was developed as a way of facilitating the communication between such synthesisers. As a result, pitch-bend, was ported to the MIDI protocol as a global variable. MIDI encodes pitch bend globally, to every note, rather than on a note by note basis. MIDI offers 16 channels, that may be used for different voicings (e.g. piano, bass, guitar). On each of these channels, there may be a certain number of notes in play at any one time. The major problem is that pitch bend is applied to the channel rather than the note. Therefore, within a given channel, you may only bend the whole group of notes, not individual notes.

### 2.2. Integrating the Seaboard

The Seaboard must produce MIDI output if it is to be easily integrated into the major existing 3rd party software programs. However, the MIDI protocol is not sufficient to represent the kinds of interaction that take place on the Seaboard. For example, using MIDI with one channel per instrument, the act of increasing the pressure or varying the location of a singular touch on the Seaboard will in fact alter the volume and pitch of all current notes being played.

This is indicative of a widespread general problem in computer music today. Namely, that it is relatively simple to create new MDCs, but it is a lot harder to integrate such systems as controllers of audio software. The next section focuses on other existing MDCs and the challenges of integration manifest for them.

## 3. OTHER MDCS

This section discusses some other existing MDCs, detailing their functionalities, the challenges they face, and their significance in the context of integration with audio software. The list is non exhaustive, and rather characterises different approaches for MDCs described later as acting, stationary and hybrid controllers.

### 3.1. The Haken Continuum

The Haken Continuum (c. 1998) is a continuous surface keyboard that offers real-time control over pitch, amplitude, and timbre [4]. The Continuum features a photoelastic playing surface lit by a single-frequency polarized light source from the underside, combining the pitch sensing and polyphonic surface abilities of the Dynamic Keyboard and the Pitch Extractor.


**Figure 2**. The Haken Continuum

The primary hardware feature of the Continuum is the continuous ribbon. In terms of MIDI integration, the Continuum faces the same problems as the Seaboard - namely that pitch bend and volume control are encoded in MIDI as global, channel parameters. The Haken team ended up offering MIDI and OSC, and developed a complex mapping software, the EaganMatrix. The Haken also has embedded sound. The difficulty of integrating their rich data with existing systems pushed them to develop an entire system in isolation from other mainstream music hardware and software.

### 3.2. Sound Beam

The Sound Beam is a single dimension controller that converts a continuous beam (a distance) into MIDI data [8]. The Sound Beam mapping, which converts an ultrasound beam into MIDI data, is a good example of an expressive conversion of continuous data to MIDI. However, the mapping is closed source, hardware dependent, and the result is monophonic since users control only one dimension, through interaction with a single beam of ultrasound. Most MDCs offer a number of dimensions, but extending beyond one immediately poses integration problems with software.

### 3.3. SoundGrasp

Sound Grasp is a gestural interface for the performance of live music that aims to facilitate musical control without the need for direct machine interaction [7]. Specifically, the system is designed to enable quick, performative recording and manipulation of audio samples. The interface takes the form of a wearable glove through which the user may perform one of eight gestures that are recognised by the system. These gestures are mapped to specific audio sampling processes such as record, play, reverb and, filter control.


**Figure 3**. Glove Positions Used in the SoundGrasp System (taken from [7])

Within the Sound Grasp system, a group of specific gestures are well-defined, and they are mapped appropriately to useful musical outputs. The system makes use of a custom and purpose-built audio processing unit for the recording and manipulation of audio. In essence, the Sound Grasp represents an example of a very well-developed framework for the mapping of gestural data to relevant musical output, but which has been developed within the sole context of audio sampling and recording.

### 3.4. Current Progress in Music Information Protocols

Progress has been made towards a more adaptable encoding protocol for musical data. For example, HD-MIDI, announced in 2005, is a proposed extension to MIDI that will bring many improvements. For example, included in the specifications for HD-MIDI is the addition of new note parameters. HD-MIDI will also bring the ability to bend individual notes [3].

The VST3 (Virtual Studio Technology) from Steinberg, launched in 1996, integrates virtual effect processors with instruments in a digital audio environment. VST3 facilitates more than one MIDI input and output at a time [2].

CopperLAN is a communication system that facilitates the interfacing and linking of professional audio and musical equipment. It features 'full auto-setup, plug and play, universal remote editing, total setup recall, streamlined and unified user interface methods, clear and concrete identification of devices' [1]. However, its major drawback is its cost, which is a barrier to adoption.

## 4. TOWARDS AN MDC FRAMEWORK

The development of the MDC framework requires the categorisation of abstract MDC gestures, and the identification of appropriate mappings from these gestures to relevant musical output. Intuitive gestures, drawing on predictable reactions experienced in the real world, should be the fundamentals of this framework. For example, in traditional musical practice, an increase in force generally results in an increase in volume, and a rapid oscillation of position generally results in a vibrato effect. The MDC framework should attempt to match high energy gestures to high energy changes in sound, and subtle movements to subtle changes in sound. MDCs not only offer control of pitch and duration, but facilitate unprecedented control over dimensions of sound such as timbre and spatialisation.

### 4.1. MDC Types

The MDC framework should take into consideration non-haptic as well as haptic controllers. As well as haptic and non-haptic, an important distinction can be made between what we call acting MDCs, stationary MDCs, and hybrid MDCs. Acting MDCs involve a device that physically moves, and outputs information about its position and orientation. A large portion of acting MDCs are non-haptic (e.g. MYO). Stationary MDCs are those that output information about a user's interaction with a device. For example, the Kinect outputs information about the user's location in 3D space, and the Seaboard outputs information about the user's touches on the surface of a physical device. An example of a hybrid MDC is a smartphone, that outputs information about the user's touches on the screen, as well as information about the orientation of the device.

The distinction between acting, stationary and hybrid is separate from the distinction between haptic and non-haptic. Any device can be either haptic or non-haptic, regardless of whether it is a stationary, acting, or hybrid MDC.

### 4.2. Data Flow

Figure 4 shows the typical data flow of a (haptic) MDC system. Firstly, in the 'Musician/Hardware' block, physical interaction takes place between user and system and this is recorded by the system's sensors. Next, in the 'MDC : Software & Firmware' block, the sensor data is collated together to extract 'events,' which are then processed and mapped to output data. In the 'Digital Music Production Software' block, data is fed into a 3rd party software sound engine, and sound output is generated (the 'Music' block). The rest of this section will discuss the 'MDC : Software & Firmware' block in more detail.

### 4.3. A Distinction of Gestural Parameters

The first step in developing a MDC framework that facilitates the integration of hardware MDCs with sound software will be the identification of a set of abstract parameters that can be grouped together to define any conceivable MDC.

Given the set of distinctions between MDC devices set out earlier in this section, we present a preliminary suggested list of parameters, grouped under 'event' (for haptic MDCs), 'object' (for non-haptic MDCs) and 'device' (for acting and hybrid MDCs). Firstly, these different parameters will be presented and then a general explanation of how gestures can be processed using these parameters will be given.

#### *4.3.1. Event Parameters*

In the case of a haptic device, user interaction with the device will usually consist of the user manipulating the device's sensors over time, through some physical, gesture-based interaction. Individual values for each sensor are stored, giving rise to data with the following structure : Sensor 1 [Value], Sensor 2 [Value], ... ,Sensor N [Value]. The data at each sensor, when taken collectively, gives rise to an instantaneous 'frame' of *events*. Each of the events in a frame have values for the following parameters :

- ID Tag
- Onset Timestamp
- Onset Magnitude/Size
- Onset X-Location
- Onset Y-Location
- Onset Z-Location

- Event Magnitude/Size
- Event X-Location
- Event Y-Location
- Event Z-Location

The ID tag is used so that specific events can be tracked over time, frame-by-frame. The 'onset' parameters relate to the state of an event during its initial frame (its initialisation). The 'event' parameters define the instantaneous state of an event, from frame to frame. The continued comparison, between some present frame and the previous frame, gives rise to specific gestures. For example, the euclidian distance in 3D space between an event's current location and its location a number of frames previously, defines a movement in a certain direction, at a certain velocity.

#### *4.3.2. Object Parameters*

In non-haptic devices, interaction typically takes place through the manipulation of identifier *objects* (e.g. user limbs/wii remote) in 3D space. Much in the same way that *events* have specific instantaneous parameters, objects have similar parameters :

- Object ID
- Object X-Location
- Object Y-Location
- Object Z-Location
- Object Orientation
- Object *State*

Objects are tracked, frame-by-frame, in a similar way that *events* are tracked. The orientation parameter is used to track rotary movements of specific objects. Relations between an object's current and previous orientation gives indication of yaw, pitch and roll gestures.

An object's 'state' refers to its configuration. Interactions with non-haptic, object-tracking systems, often include object-specific gestures, where objects go from being in one state/configuration to another. For example, an open hand may close into a fist, which would be recognised by the system as a meaningful gesture. The specific possible configurations of objects will depend on the specific system, and the identifier objects that are used. For example, in the SoundGrasp system the objects are the user's hands, and the hands can be in any of 8 specific 'states' (see Figure 3).

#### *4.3.3. Device Parameters*

For acting and hybrid MDCs, parameters are required that describe the location and orientation of the device itself. These can be defined as :

- Device Orientation
- Device X-Location
- Device Y-Location
- Device Z-Location

Device-specific gestures can be tracked by tracking the differences in these parameters from frame to frame.

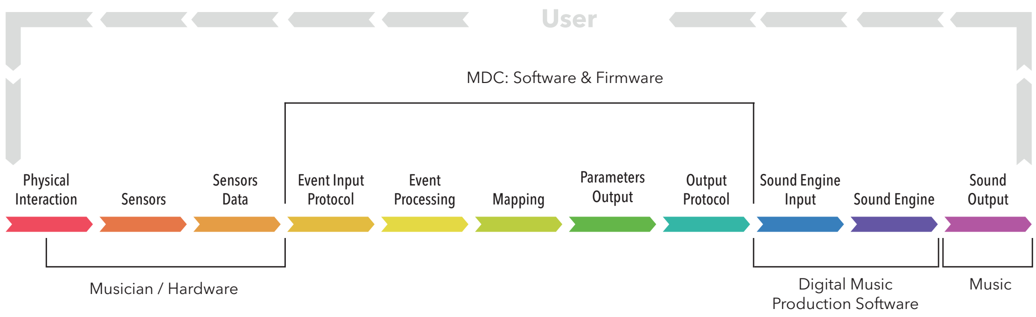


**Figure 4**. Data Flow in a Stationary Haptic MDC

## 4.4. Mapping

Having set out a list of parameters that may be used by different types of MDCs to track gestures as they occur, the next key step in facilitating effective gestural control of sound is to define effective mapping strategies such that gestures produce relevant, intuitive sound output. The design of an appropriate and effective mapping strategy is one of the most important issues in the development of new MDCs. This issue of mapping is influenced by three key factors : real-time constraints, dimensionality, and customisation.

### *4.4.1. Real Time Constraints*

MDCs need to work in real-time, with minimal latency between user action and resulting sound output. In order to achieve minimal latency, the number of previous frames kept in memory must be small - usually only 1 or 2. This means that as gestures occur, systems only have access to instantaneous moment-by-moment segments of them.

This is an important restriction, given the 'temporal precision' with which musical performers can perform gestures. Schmeder et. al (2010) give an explanation of the term temporal precision [9]. Temporal precision refers to the smallest time interval over which humans have control in terms of musical gestural performance. Using the example of the 'flam' technique from drumming, Schmeder et. al report this smallest time interval to be 1 millisecond.

Therefore, in order for gestures to produce meaningful sound output, systems must make use of relevant mapping strategies between event parameters and resulting output data. For example, by mapping instantaneous x-location to pitch, smooth movements over the x-axis will effectively translate as smooth glides in pitch.

### *4.4.2. Dimensionality*

Mapping strategies can exist as one-to-one connections, one-to-many connections, or many-to-many connections. Hunt et. al (2002) performed an experiment to evaluate the effects that different types of dimensionality mappings had on users [5]. They found that users learned the performative limits of one-to-one mappings very quickly, but that this lead them to perceive a sense of restrictiveness in the system. Conversely, many-to-many mappings were harder to learn, but this gave users a sense of performative potential that they could spend time mastering.

### *4.4.3. Mappings and Customisation*

Although parameters must be mapped to relevant sound outputs, there is a question as to whether or not all mappings should be permanent.

Permanent mappings help to establish MDCs with their own performative identity, and can help foster intimate relationships between user and system. In the realm of musical instruments, it is through the struggle with an instrument's inherent restrictions (and permanent mappings) that true creative innovation occurs.

However, one of the great advantages of new MDC systems is the level of personalisation and customisation they facilitate. Since there is no physical connection between gesture and sound output, users could be free to map *any* gesture to *any* sound output. Indeed, this could potentially open up a whole new paradigm for user creativity with digital musical instruments. As well as exploring the new performative potential offered by digital musical interfaces, users could start creating and sharing new performative possibilities, by experimenting with user-defined mappings. An analogy in the realm of traditional musical instruments is the use of a violin bow to bow the strings of an electric guitar. Allowing users to experiment with new mappings in the digital domain should be facilitated at the front end of MDCs, and shouldn't require an understanding of the whole protocol chain. Detaching the mapping discovery from the constraints of low level physical and digital inputs and outputs is integral to our proposition.

## 4.5. Output Data - Beyond MIDI and OSC

There are specifications available that could be used to encode the MDC state and gestural controls in terms of musical parameters. For example, MIDI or OSC. The main problems with MIDI have been described many times before, but it appears useful to rephrase them here in light of the MDC requirements. Firstly, MIDI is based on the piano paradigm in which notes exist as discrete events with on/off values. Secondly, MIDI encodes pitch bend and volume as global parameters, rather than on a note by note basis.

OSC addresses the limitations of MIDI in many ways : it is high definition, low latency, flexible and multi-dimensional. Schmeder et al. (2010) suggest to use ontology-oriented description of events, and suggest that the naming of these events should originate from the application designers [9]. But overall, the lack of a prescribed ontology for OSC messages, and the requirements at the physical level (Ethernet), appear to be obstacles to a mainstream adoption. We argue that, for a music protocol to gain adoption, it has to offer a common definition of discrete and continuous events, notes, volumes, pitches and durations and also timbre or spatialisation.

#### *4.5.1. Note Events*

The note events consist of the data that is visible to the sound engine. In contrast to the *gestural event* parameters described earlier, the *note event* parameters are a result of the mapping and possible customisation happening at the hardware and firmware levels. For the MDC framework to overcome MIDI shortcomings, it has to offer polyphonic control over individual note parameters, during the lifecycle of notes. Within each individual note event, changes in timbre, pitch and volume need to be available in real time and high definition to the sound engine. The definition of these note events will progress through consultations with audio software developers, artists and pro users.

Given the groundwork already covered by the MIDI protocol, an important question in the development of the MDC framework is whether the MIDI specifications should be extended to allow for richer data, or whether a whole new protocol should be developed. More fundamentally, we argue that the MDC framework, possibly in the form of an API, should aim at transcending the transport layer (USB MIDI, TCP/IP or Bluetooth) to focus on facilitating the end to end communication with comprehensive and meaningful gestures to produce rich, expressive, polyphonic sounds.

### 4.6. Framework Requirements

The success of this new software protocol design relies on collaboration between its three primary user groups : artists (end users), hardware manufacturers, and audio software developers.

In order to gain support and participation for this new framework within (a) hardware manufacturing, (b) software developers and (c) artist communities, an open approach must be taken towards IP and cost. Developers and manufacturers will only be willing to get on board with the framework if they can implement it openly and freely within their products. The inability to integrate novel inventive MDCs with existing software packages is a major problem, and the only way to reach a widespread, sustainable solution is to make that solution open-source, adaptable, and understandable.

The end users of MDCs and audio software will come into varying levels of contact with the framework, depending on their interest in configuration, personalisation and experimentation. The framework, therefore, needs to be easily readable and usable to end users of music hardware and software.

## 5. CONCLUSION

This paper has outlined the difficulties in integrating multi-dimensional controllers (MDCs) with audio software programmes, through the specific example of the development of the Seaboard and its integration using the MIDI protocol. A proposal has been put forward for a new MDC framework that focuses on gestural control and relevant musical output. The key issues involved in the development of this framework have been discussed.

The goal of this proposal is to state ambitious goals rallying the communities of new instrument makers, audio software developers and musicians eager to bridge the existing gaps between hardware and software worlds, and produce more meaningful, intuitive controls for sound production. The proposed MDC protocol would provide a common language for developers and manufacturers to quantify, discuss, share and integrate their gestural controls and mappings. The development of this new protocol will be best facilitated by open discussion and participation by those currently invested in the development of MDC systems.

## 6. ADDITIONAL AUTHORS

*Roland Lamb*
ROLI
roland@weareroli.com

## 7. REFERENCES

[1] http ://www.copperlan.org/index.php.

[2] Http ://www.steinberg.net/en/company/technologies/v.

[3] C. Anderton. MIDI : The Next 30 Years. *Electronic Musician*, page 12, Feb. 2013.

[4] L. Haken, R. Abdullah, and M. Smart. The continuum : a continuous music keyboard. In *International Computer Music Conference*, page 81, 1992.

[5] A. Hunt, M. M. Wanderley, S. S. West, and M. Paradis. The importance of parameter mapping in electronic instrument design. In *New Interfaces for Musical Expression (NIME)*, pages 1–6, 2002.

[6] R. Lamb and A. N. Robertson. Seaboard : a new piano keyboard-related interface combining discrete and continuous control. In *New Interfaces for Musical Expression (NIME)*, pages 503–506, 2011.

[7] T. Mitchell and I. Heap. SoundGrasp : A Gestural Interface for the Performance of Live Music. In *New Interfaces for Musical Expression (NIME)*, number June, pages 465–468, 2011.

[8] S. Paul and D. Ramsey. Music therapy in physical medicine and rehabilitation. *Australian Occupational Therapy Journal*, (47) :111–118, 2000.

[9] A. Schmeder, A. Freed, D. Wessel, A. Street, and B. Ca. Best Practices for Open Sound Control. In *Linux Audio Conference*, 1997.