TC-Data: Extending Multi-Touch Interfaces for Generalized Relational Control

Kevin Schlei
Department of Music
Peck School of the Arts
University of Wisconsin-Milwaukee
Milwaukee, WI
kdschlei@uwm.edu

ABSTRACT

This paper presents TC-Data, an OSC and MIDI controller application for iPad that uses multi-touch relational control. The application is designed to allow for customization of controller data streams and message formatting. This paper discusses the design of the programming interface, as well as the control paradigms that best fit the application's functionality. The concept of a convoluted control state is considered as a guide for programming target systems.

1. INTRODUCTION

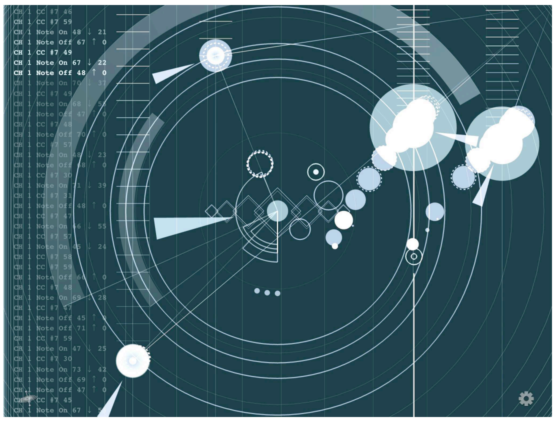
This research extends earlier explorations of relationship-based (or relational) data streams driven by multi-touch performance. Prior work focused on realizing self-contained environments where audio synthesis was the primary goal [1][2]. These systems, even when built around a modifiable programming interface, still confined the relational controllers to a fixed synthesis graph.

Other systems of gathering multi-point data [3][4] provide invaluable raw performance data for musicians and artists to incorporate into their works. These data streams can be brought into audio or visual programming environments such as Pd or Processing to drive a customized performance. Users of these protocols often significantly process the raw data to gain useful data streams. By adding a layer of relational data analysis to the raw data prior to sending out generalized messages, users can benefit from the creative data being generated by multi-touch performance without taking added steps.

2. RELATED WORK

Many of the pioneers of multi-touch research created generalized interfaces capable of OSC or MIDI messaging. The touch data from Davidson and Han's large scale multi-touch display was transmitted to the host system via OSC to drive a multitude of activities [5]. The TUIO protocol is a flexible and widely used system for the communication of multi-point data to host systems [3]. Flat surfaces can be transformed into multi-touch interfaces that report X/Y coordinates in addition to touch amplitude [4]. Systems like these provide raw performance data that is then mapped to synthesis parameters.

The Lemur and Konkreet Performer projects provide self-contained control environments where the sensing surface and backing software is contained on the same hardware device [6][7]. These applications use interactive on-screen elements, including movable control points, skeuomorphic sliders, and knobs.



```
CH 1 CC #7 46
CH 1 CC #7 59
CH 1 Note On 48 ↓ 21
CH 1 Note Off 67 ↑ 0
CH 1 CC #7 49
CH 1 Note On 67 ↓ 22
CH 1 Note Off 48 ↑ 0
CH 1 Note On 70 ↓ 37
CH 1 CC #7 49
CH 1 Note On 68 ↓ 55
CH 1 Note Off 47 ↑ 0
CH 1 CC #7 48
CH 1 Note Off 70 ↑ 0
CH 1 CC #7 57
CH 1 Note On 48 ↓ 23
CH 1 Note Off 68 ↑ 0
CH 1 CC #7 30
CH 1 Note On 71 ↓ 39
CH 1 CC #7 31
CH 1 Note Off 48 ↑ 0
CH 1 CC #7 47
CH 1 Note On 66 ↓ 55
CH 1 CC #7 57
CH 1 Note On 45 ↓ 24
CH 1 CC #7 58
CH 1 CC #7 59
CH 1 Note Off 66 ↑ 0
CH 1 CC #7 48
CH 1 Note On 69 ↓ 28
CH 1 CC #7 47
CH 1 Note Off 45 ↑ 0
CH 1 Note Off 71 ↑ 0
CH 1 CC #7 59
CH 1 Note On 47 ↓ 25
CH 1 CC #7 30
CH 1 Note On 73 ↓ 42
CH 1 Note Off 69 ↑ 0
CH 1 Note Off 47 ↑ 0
CH 1 CC #7 45
CH 1 Note On 67 ↓ 54
```

Figure 1. The performance screen of TC-Data. Graphic representations of touch controllers, such as speeds, distances, positional relationships, and timings help inform performance.

TC-Data is based on a prior synthesizer application TC-11 [2]. The goal of TC-Data is to liberate the relational controllers by organizing them and broadcasting them via the common messaging protocols Open Sound Control (OSC) and MIDI. Some possible non-musical uses include visual control, lighting control, physical computing control, and multi-touch performance analysis.

3. DESIGN

3.1 Performance and Relational Controllers

Like its predecessor TC-11, TC-Data is built around the iPad's multi-touch screen (Fig. 1). User touches generate raw multi-touch information, which is analyzed to create relational data. The iPad's motion sensors are used to follow physical motion in performance.

In the context of this paper, the term *controller* will be used to describe a relational data stream generated by the multi-touch performance. For example, the Group Total Speed is a controller generated by summing each touch's speed across the performance area.

The term *trigger* is used to describe a particular type of controller that is a single temporal event. For example, Roll Crossed Center is a device motion trigger that fires when the user crosses the zero point of the gyroscope roll axis. When tilting the device back and forth, a trigger is fired each time the axis is crossed.

In total there are 154 multi-touch controllers, 62 device motion controllers, and 88 multi-touch and device motion triggers.

3.2 Programming Basics and Terminology

Programming in TC-Data centers around the creation of *outputs*. Outputs are objects that patch controller streams through to a target destination. Outputs translate the controller stream to match the value range and expected format of the target input. To create a TC-Data patch, users add outputs and assign controllers, value ranges, and sensitivity adjustments to them. During performance, controllers feed the outputs, which translate the incoming values and send them to the target.

Each output has a *target*, which is the external destination for patched controller values. For OSC outputs, the target will have an OSC Type Tag String, which will tag each message sent through the network to the IP address and port defined in the application settings. For MIDI outputs, the target is a MIDI port and channel. Continuous controller targets also include the controller number.

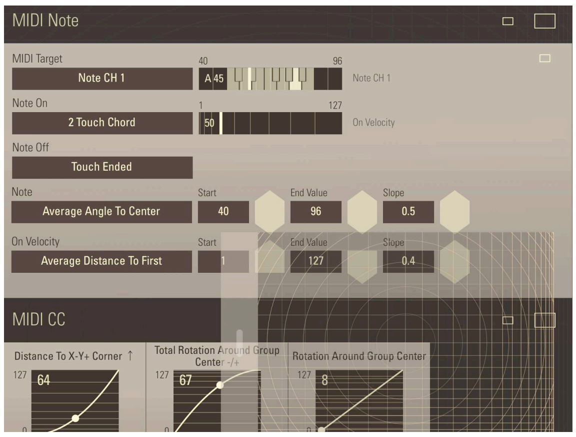

MIDI Note
MIDI Target: Note CH 1 — 40 / 96 — A 45 — Note CH 1
Note On: 2 Touch Chord — 1 / 127 — 50 — On Velocity
Note Off: Touch Ended
Note: Average Angle To Center — Start 40 — End Value 96 — Slope 0.5
On Velocity: Average Distance To First — Start 1 — End Value 127 — Slope 0.4
MIDI CC
Distance To X-Y+ Corner ↑ — 127 — 64 — 0
Total Rotation Around Group Center -/+ — 127 — 67 — 0
Rotation Around Group Center — 127 — 8 — 0

Figure 2. A MIDI Note Output. Note on / off triggers, note number, and on velocity parameters are set here.

Outputs contain one or more *parameters*. A parameter represents a controllable or trigger-able value, which has a defined range limit. For example, a MIDI CC Output has one parameter: the continuous controller value. A MIDI Note Output has six: the note on and note off trigger parameters, note number parameter, on velocity parameter, and optional off velocity and polyphonic aftertouch parameters.

While the original intent of the multi-point relational controllers was to manipulate audio synthesis, the terminal destination of the messages sent from TC-Data can be any OSC or MIDI compliant system. For this reason, the term *target system* will be used to describe the device receiving the outgoing messages. Some examples include a MIDI synthesizer, audio programming environment (Pd, Max/MSP, Chuck), visual programming environment (Processing, Jitter), lighting system, or physical computing system.

3.3 Outputs

The most basic outputs are the OSC Controller and OSC Trigger outputs, which take their control source and send out OSC formatted network messages. The message list consists of the value being sent and its voice assignment. More information on voice allocation will be discussed in 3.5.

MIDI outputs translate controller streams to the 7 or 14-bit value ranges expected by the MIDI protocol. Simple MIDI message types such as continuous controllers, pitch bend, and channel aftertouch take a single controller source and send out a translated MIDI message. Compound MIDI messages types, such as program changes with bank changes, and note messages, require a combination of controllers and triggers to perform in a specific order.

The MIDI Note Output (Fig. 2) presented a design challenge. A completed MIDI note consists of up to six variable elements: a note on trigger, note value, on velocity value, note off trigger, off velocity value, and polyphonic aftertouch value(s).[1] In a closed-system environment the generation of each of these elements can be in a fixed pipeline, but with each element separately controllable in TC-Data a number of issues arose.

First, the symmetry of note on / note off messages should generally be maintained to avoid "stuck notes." In a traditional interface, keyboard keys or buttons solve this issue by physically toggling between on and off, forcing a reset. By allowing any trigger to be used for note on / note off events, TC-Data cannot guarantee that symmetry will be achieved. For example, a note triggered on by a device motion-based trigger has no inverse or reset event to turn off the note. For programming flexibility, no limits are set on which triggers can be used in conjunction with one another. Users may design a target system that only requires note on messages. In this case, using no trigger at all for note off messages is an option. For cases where a note off is required, but no available trigger can provide a useful solution, a special note off trigger was created as a catch-all. The trigger simply sends a note off message 125ms after the note on message is triggered.

Second, the note number and on velocity parameters are unique in that they are only set once during the lifecycle of the note. This is in contrast to a continuously changing MIDI message such as pitch bend. Certain controllers begin their values at the same point each time.

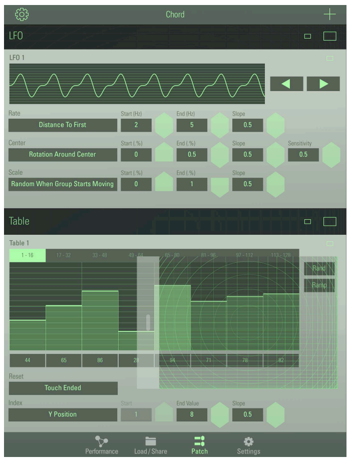
[1]While polyphonic aftertouch messages are technically independent of note messages, they are often tightly coupled. For example, polyphonic aftertouch values are sent on physical keyboard controllers if their key is activated, which sends a note on message. For this reason the MIDI Note Output was chosen as the method of sending polyphonic aftertouch rather than making it a separate output type.

For example, the Distance to Starting Position controller follows how far the touch is from its original starting location on screen. The first value for this controller is always zero, making it a poor choice for a note number or on velocity parameter. Because parameter control was designed around a universal programming system, these options are available to the user even if they are not effective.

3.4 Modules

There are four internal modules available for augmented controller generation. They are the AHDSR (envelope generator), LFO (low frequency oscillator), Table (array lookup), and Sequencer (step sequencer). These modules become controllers themselves which can be assigned to any other output, module, or even their own parameters.

An example of how a module is used to augment generated controllers is the Table (Fig. 3). The Table supplies an indexed array of up to 128 values. Its index parameter, like other parameters, can be assigned to follow any controller. That controller moves the Table index position, which reads and outputs the array value. The goal of the Table is to split any controller into discrete values. Smoothly ramping touch or device motion controllers can be quantized into user-defined values. If a Table has a length of two values, this can create a controller action similar to the MIDI “'switch” continuous controllers.



Rate: Distance To First; Start (Hz) 2; End (Hz) 5; Slope 0.5
Center: Rotation Around Center; Start (.%) 0; End (.%) 0.5; Slope 0.5; Sensitivity 0.5
Scale: Random When Group Starts Moving; Start (.%) 0; End (.%) 1; Slope 0.5
44 65 86 28 94 71 78 82
Reset: Touch Ended
Index: Y Position; Start 1; End Value 8; Slope 0.5

Figure 3. LFO and Table modules.

In addition, the Table sends a trigger event whenever a new index is read. This means that any controller can be split into a set of triggers dividing its value range.

It is worth examining the roll of modules in TC-Data. Their primary functionality is to create useful common control streams. For example, the AHDSR can create a responsive, changing envelope shape that can fade quickly in some touch motions and slowly in others via its parameter controller assignments. While a user could design their own AHDSR module (or other) using the outputs provided, the internal modules provide a ready solution.

The greater roll they serve is as a microcosm of the type of programming that TC-Data serves to facilitate. An ideal scenario for TC-Data is to send many data streams which, when assigned to control a variety of target system parameters concurrently, provide a creative and complex interface. The modules follow this paradigm by housing discrete parameters controllable by the same system as the rest of the output parameters.

3.5 Voicing

Like its predecessor TC-11, TC-Data separates its controllers and triggers into two categories: *voiced* or *global*. Individual touches represent a single polyphonic voice. Groups of touches combine together to represent a global value.

Individual touches cycle through the available voice allocation slots. New touches are assigned the first available voice index, and removed touches release their index for future assignment. The maximum number of touches is 11, matching the iPad hardware limitation.

The voice index numbering differs from the TUIO [3] voice allocation system. Rather than incrementing a unique touch identification number, TC-Data opts for the cyclical allocation system described above. This system allows TC-Data to work well with polyphonic synthesis graphs. However, the design prevents it from maintaining persistent value storage for individual voices. It also means that the assigned voice index is unpredictable from the performer's perspective. Custom target systems should be designed with this in mind: the 11 available voices should generally not be hard assigned to specific actions.

A major issue with voicing is the loss of data when translating voiced controllers to the MIDI protocol. While custom target systems using OSC can route voiced controllers to their individual targets, continuous parameter data in the MIDI protocol is channel data.[2] While each touch could theoretically have its own MIDI CC values, only one is sent to the assigned MIDI channel to avoid zippering. The solution to this issue would be channelizing of voiced data to multi-timbral MIDI targets, but is not currently implemented.

4. USE SCENARIOS

4.1 Standalone MIDI Controller

TC-Data can create all MIDI message types except System Exclusive messages. Patches can trigger synthesizers in a traditional manner, using MIDI Note Outputs and any combination of controller outputs. Using the Sequencer module, melodic and rhythmic patterns can be performed. The MIDI Note Output supports filtering notes into custom scales. A note filter parameter allows real-time switching between 12 filters.

2 Polyphonic aftertouch is the exception, as it is tied to a specific note value in addition to its MIDI channel.

4.2 MIDI Controller Augmentation

TC-Data makes little effort to replicate the design of a keyboard for note triggering. While grids can be placed on screen that read note values from a Table module, they cannot be considered a serious replacement for a physical keyboard.

It may be fruitful to use TC-Data in conjunction with a hardware keyboard MIDI controller (Fig. 4). The idea is to play to the strengths of each of interface. The keyboard controller is a tested, successful note creation interface. TC-Data could feed continuous controller, pitch bend, aftertouch, etc., to the same destination to provoke an augmented response.

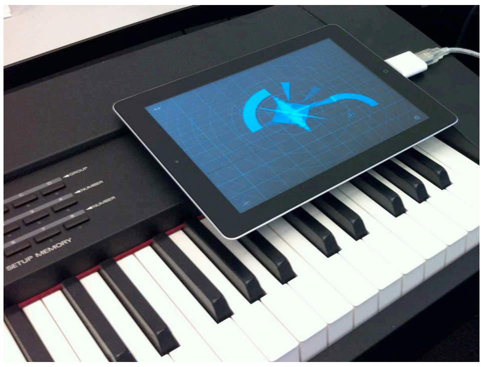


Figure 4. TC-Data running on an iPad, connected to an external MIDI interface. The MIDI controller data can be merged with the keyboard output.

4.3 Multimedia Performance Tool

Sensor driven multimedia could be designed around TC-Data's output data streams. Generative video and animation could be played with the same gestures that drive a musical performance.

4.4 Touch and Motion Analysis

Data streams could be stored and analyzed to extract performance information. Users interested in multi-touch techniques could find use in collecting touch information from the relational controllers, rather than simply from the coordinates.

Development of a notation system for multi-touch performance could benefit from data analysis of user performances to determine how accurately a notated phrase was executed. Rather than attempting to match raw coordinate data with expected results, more focused questions could be asked, like “Did the user open their hand wide enough?”

5. DISCUSSION

5.1 Tangible or Intangible?

We can consider an interface as tangible if it has physical, manipulatable, representational, and spatially reconfigurable qualities [8]. An intangible interface consists of interacting with information represented by pixels on a display, or Graphic User Interface (GUI) [9]. At first glance, we could view TC-Data as intangible. Performance exists in the virtual space of the display, as reflected by the graphics drawn to inform the performer of the controllers used.

However, there are no interactions with on-screen artifacts. No virtual objects are manipulated, no buttons pressed, no information accessed and prodded. The visual display is not a gateway to interaction. It is redundant: a helpful tool for performative feedback, but not a required component.

Many of the relational controllers are generated through touch motions and relationships independent of their screen position. For example, the distance between two touches is independent of their position on the X/Y plane. If the edges of the screen could be extended infinitely, these controllers would be unaffected by the lack of a defined coordinate space. Practically this frees the performer from orienting themselves within the screen's bounds, allowing them to perform the instrument without looking at it. This quality is often found in tangible interfaces, and rarely in touch screen interfaces.

TC-Data allows the user to set the display (Fig. 1) to draw specific controllers. The user can also choose to have no controllers drawn on screen, effectively blanking the display. Performance tests have shown that once a user has explored a particular set of controllers in a patch they no longer need the visual feedback to perform effectively.

There are exceptions to the coordinate independence of relational controllers. Patches can divide the screen into discrete sections and grids. Like a picture of a piano keyboard, these grids can become an interactive visual artifact.

Is TC-Data tangible or intangible? It is a tangible interface, with optionally added intangible qualities. The iPad hardware is treated as a tangible object. By design the screen is touched from edge-to-edge, with the bare minimum of user navigation objects on the screen, and often without the requirement of a visual display. The iPad is physically moved in space to generate control data. The nature of the relational controllers and design of the performance interface strongly suggests it meets the standards of a tangible interface.

5.2 The Convoluted Control State

Controller data streams in TC-Data exist in complex relationships with one another, often in flux. One way to view the complexity is to consider each controller an axis of performance. Like the X and Y axes of a 2D space, which can be simultaneously but separately traversed, many controllers can maintain independence from one another even when being manipulated by the same group of touches [1]. However, they can also be strongly linked to one another, in which case it is very difficult to manipulate one without affecting others.

This state of complexity, where a group of touches can both independently and coherently alter collections of controllers can be considered a *convoluted control state*. Once the controller density reaches critical mass, the ability to discretely alter a single controller without disturbing any others becomes impossible.

This is a desirable place to be when aiming for multi-dimensional, expressive, or otherwise complex performance results. In interfaces with single function UI elements, such as an array of physical knobs, the more elements added the more impossible it becomes to manipulate them in concert. The opposite is true for a convoluted control state. At any point in the phase of a convoluted control state, parameter control is organized in a complex but ordered system of relationships.

Design of the target system should reflect the idea of the convoluted control state. TC-Data will serve poorly as a replacement mixer front-end, where it is desirable to maintain individual control of the states of discrete elements. For example, using TC-Data as a surround panner for 11 unique sounds may prove difficult, due to the rotating polyphony assignments. State storage should be handled by the target system. In the surround panner example, the software could follow a particular output of TC-Data to lookup and connect a particular sound clip to the live activity, and hold its value when not active.

If target systems are designed with the convoluted control state in mind, the performative aspect of TC-Data should be an attractive front end to a number of otherwise static, discretely manipulated, or automated environments.

6. CONCLUSIONS

6.1 General Conclusions

This paper outlined the design, implementation, and functionality of TC-Data, an application created to provide relational controller data in a generalized format. Considerations of the nature of the interface provided a conceptual framework for incorporating the application into a performance system. The idea of a convoluted control state may assist in the creative development of use scenarios.

6.2 Issues and Future Work

One of the major hurdles to overcome is the data bandwidth and latency of output messaging. OSC message must deal with wireless network limitations. OSC messages are sent out at higher frequency and in greater numbers than MIDI messages. This is partly because of the increase in resolution from 7 to 32 bit values. A single touch controller can generate approximately 40 kB/second of network data after thinning methods have been applied. While Wi-Fi networking has plenty of bandwidth for general use cases, it is clear that a limit can be reached. A more pressing concern is latency and interrupted data flow introduced in network traffic and wireless transmission. A wired network solution would be best, but is currently unavailable through the iOS SDK.

MIDI messages have a latency issue due to the standard MIDI transmission rate of 31.25 kb/second. While an approximately 1 ms message rate is sufficient for most basic MIDI control, TC-Data has a problem due to the frequency of controller value changes, and the order in which they are sent. All controllers evaluate and send before triggers, meaning note messages are always last in the chain. This means that every note message will be preceded by all other MIDI message types, adding milliseconds of delay to each note.

As mentioned in 3.5, polyphonic MIDI control could be possible through channelized voice assignments, and should be implemented in future versions.

Acknowledgments

Special thanks to the Composition and Technology faculty at UWM and the members of the TORCH ensemble for their invaluable input and support.

7. REFERENCES

[1] K. Schlei “Relationship-Based Instrument Mapping of Multi-Point Data Streams Using a Trackpad Interface,” in *Proc. of the Int. Conf. on New Interfaces for Musical Expression*, Sydney, 2010, pp. 136-139.

[2] K. Schlei, “TC-11: A Programmable Multi-Touch Synthesizer for the iPad,” in *Proc. of the Int. Conf. On New Interfaces for Musical Expression*, Ann Arbor, 2012.

[3] M. Kaltenbrunner, T. Bovermann, R. Bencina, and E. Costanza, “TUIO - A Protocol for Table Based Tangible User Interfaces,” in *Proc. of the 6th Int. Workshop on Gesture in Human-Computer Interaction and Simulation*, Vannes, 2005.

[4] A. Crevoisier, and G. Kellum, “Transforming Ordinary Surfaces into Multi-touch Controllers,” in *Proc. of the Int. Conf. on New Interfaces for Musical Expression*, Genova, 2008, pp.113-116.

[5] P. L. Davidson, and J. Y. Han, “Synthesis and Control on Large Scale Multi-Touch Sensing Displays,” In *Proc. of the Int. Conf. on New Interfaces for Musical Expression,* Paris, 2006, pp. 216-219.

[6] http://www.jazzmutant.com/

[7] http://www.konkreetlabs.com/

[8] B. Ullmer, Tangible Interfaces for Manipulating Aggregates of Digital Information, *Doctoral*, 2002.

[9] H. Ishii, “Tangible User Interfaces” in The Human Comptuer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications, 2nd ed. Sears, A., Jacko, J. Ed. CRC Press, 2008.