TC-11: A Programmable Multi-Touch Synthesizer for the iPad

Kevin Schlei
University of Wisconsin-Milwaukee
3223 N. Downer Ave.
Milwaukee, WI
kevinschlei@gmail.com

ABSTRACT

This paper describes the design and realization of *TC-11*, a software instrument based on programmable multi-point controllers. TC-11 is a modular synthesizer for the iPad that uses multi-touch and device motion sensors for control. It has a robust patch programming interface that centers around multi-point controllers, providing powerful flexibility. This paper details the origin, design principles, programming implementation, and performance result of TC-11.

Keywords

TC-11, iPad, multi-touch, multi-point, controller mapping, synthesis programming

1. INTRODUCTION

The iPad introduced incredible potential for mobile music creation with its large multi-touch screen [3]. The transition from sensor to "painterly interface" means accessing the intrinsic qualities of the multi-point data, while working within the extrinsic confines of the sensor capabilities [4].

Seamless interactive experiences have been created using virtual interface elements and physical sensors in mobile devices. These "tangible artifacts," such as the a virtual violin *Magic Fiddle* [7], blur the distinction between sensor interaction and symbolic reality.

TC-11 seeks to avoid a "direct manipulation" interface design, where visual metaphors are explicitly represented and operated on [2]. No virtual objects or widgets are incorporated for performance. Instead, the user's multi-touch performance is directly connected to the synthesis engine.

Previous research by the author [5] looked into effective ways to use multi-point data streams for synthesis control. While the controller functionality was successful, the implementation was fragmented between multiple applications, and lacked a unified synthesis pipeline. The goal of this project was to join multi-point performance with programmable synthesis parameters.

In this paper, the term *multi-point controllers* refers to control streams created by analyzing the raw multi-touch information from the iPad. *Synthesis objects* are high level unit generators that can have multiple *parameters* for manipulation. *Control modules* are controllers created in the synthesis graph (envelope generator, sequencer, and low frequency oscillator).

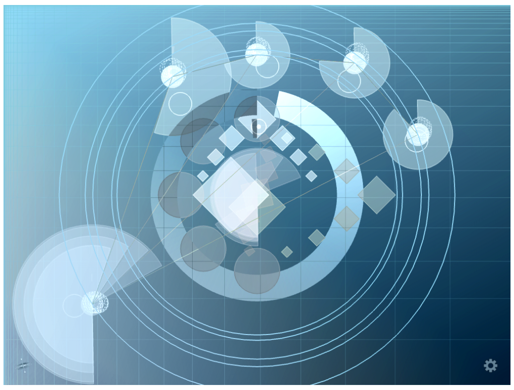


Figure 1: The performance area displays visual representations of multi-point controllers.

2. DESIGN PRINCIPLES

The design target was to provide a deep multi-touch performance interface, with a programming environment that could flexibly connect multi-point controllers to synthesis objects. Traditional synthesis interfaces, such as keyboards, sliders, knobs, or buttons, were intentionally excluded from the performance area. This required the multi-point control implementation to be robust, and focused performance on multi-touch interactions rather than widget manipulation.

2.1 Programming Paradigm

The programming environment was built around the idea that any synthesis parameter should be controllable by any multi-point controller. The starting point was building a collection of multi-point controllers to generate control data from the touch screen. The design challenge was creating a system where the multi-point controllers were adjustable, but also easily programmable into the synthesis graph.

When programming *MStretchSynth* and *MDrumSynth* [5], early multi-point controlled synthesizers, much of the work focused on scaling and mapping of incoming controller ranges. Sensitivities of the controllers needed to be adjusted to fit the desired synthesis result. This functionality had to be included in TC-11 to provide a versatile programming environment.

The user would program synthesis parameters by setting their control sources and value ranges. In some cases, the controller sensitivity could also be adjusted. Users would not have the ability to create their own multi-point controllers from scratch.

3. IMPLEMENTATION

3.1 Hardware and Software

The iPad was chosen for its single-device integration of multi-touch hardware, programming tools, and on-board audio production.

The iOS multi-touch functions calculate changes in the user's performance asynchronously. The touch data is updated when the user begins a new touch, moves a touch, ends a touch, or cancels a touch. The multi-point controller values are calculated within these function calls. This is an improvement over a previous implementation [5], where relationship-based analysis was linked to the graphics' frame updates, regardless of multi-point activity.

libpd [1] was chosen for the synthesis engine. In general, control data is generated within the iOS functions and sent via messages to the Pd audio graph, where it controls the synthesis objects. The exception is the implementation of control modules, which is entirely located within the audio graph.

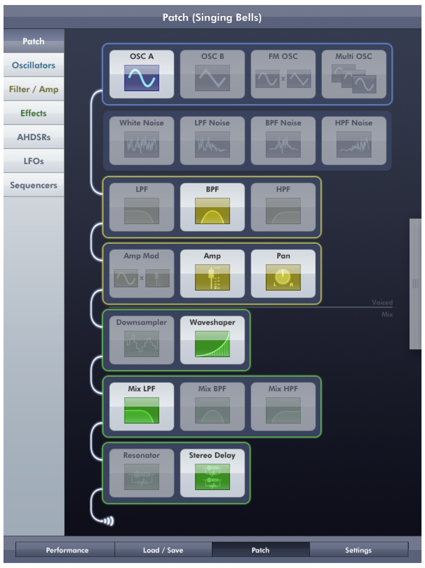


Figure 2: Synthesis objects can be toggled on or off during patch creation.

3.2 Controllers

3.2.1 *Multi-Point Controllers*

Controllers are created from the multi-touch interface using relationship-based analysis [5]. There are two types of controller: continuous controllers and triggers. Continuous controllers target any parameter with a variable value. Triggers target module actions such as *Start*, *Stop*, *Reset*, etc..

Each type of controller is further split into two categories: *Single Touch* or *Group Touch*. Single Touch controllers are controllers whose data source is tied to an individual touch. For example, *Touch Distance To Center* calculates each individual touch's distance to the center point of the screen. Group Touch controllers get their data by analyzing the collection of touches. *Total Group Speed*, for instance, is the sum of all the touches' speeds. The same categorization is applied to triggers.

3.2.2 *Device Motion Controllers*

The iPad's accelerometer, gyroscope, and compass are implemented as controllers. They appear along side multi-point controllers when programming, and use the same patching system.

3.2.3 *Control Modules*

Some traditional control modules were added to TC-11 to make patch programming easier. These modules are: envelope generators, step sequencers, and low frequency oscillators. Each module has a number of parameters that are controlled via the matrix patching system used for the synthesis parameters.

Modules are intended to assist with common synthesis control tasks. For example, an envelope generator often controls the amplitude of each synthesis voice. Step sequencers, when targeting certain frequency parameters, can map their values to MIDI note frequencies.

3.3 Synthesis

TC-11 has a set of modular synthesis objects. Objects include wavetable oscillators, filters, amplitude controls, and effects. Each object has a set of synthesis parameters. For example, the *OSC A* object has a selectable waveform, frequency parameter, and level parameter. These parameters are directly targeted by controllers, and setting their variables is the primary method of patch programming.

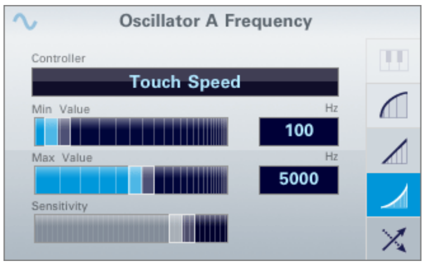
The desired number of polyphonic voices was 11, which would match the maximum simultaneous touches on the iPad's screen. However, the audio processing performance was not sufficient enough to handle this, so polyphony was limited to 8 voices.

3.4 Matrix Patching

Connecting multi-point controllers to synthesis objects was one of the main technical hurdles of the project. Three capabilities were planned to provide the most control from the multi-point controllers:

1. The ability to control any synthesis parameter with any controller
2. The ability to define each parameter's value range
3. The ability to set the sensitivity, direction, and slope of the incoming controller

To accomplish these goals, a matrix patching system was implemented that connects controller data to synthesis parameters. Each controller is allocated an array to hold its parameter targets. When the controller updates its value, it iterates through the array and sends a normalized value to "patcher" objects, which map the value to the parameters' ranges and slopes. Finally, the patched value is sent to the actual receiver in the audio graph (Figure 3).


Oscillator A Frequency

Controller
Touch Speed

Min Value Hz
100

Max Value Hz
5000

Sensitivity

Figure 4: The standard synthesis parameter view.

Control modules perform their functions inside the *libpd* audio graph in order to take advantage of the timing control inherent in Pd. This presented an issue with the matrix patching system, since the functions for routing and patching controller values resided in the iOS portion of the program. These functions had to be duplicated inside the audio graph in order for modules to coexist with the other controller types.

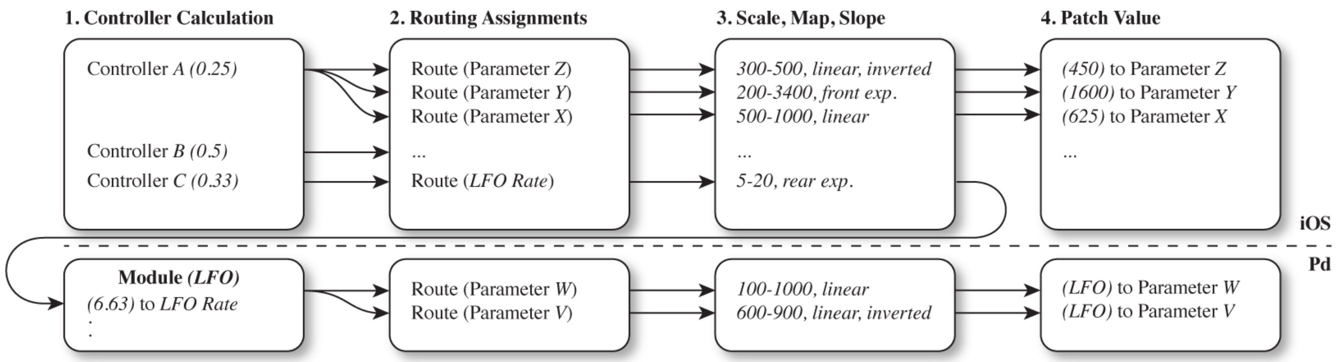


Figure 3: A diagram showing the patching flow from controller to synthesis parameter.

One advantage of this matrix patching system is the separation of the controller, parameter, and value mapping. This makes adding additional controllers or synthesis objects trivial, since no firm connections between the controllers and audio graph exist.

The most significant benefit is the ability to use one controller to manipulate multiple synthesis parameters. This allows for a single controller to generate a complex synthesis reaction, since each parameter sets its own individual response to the incoming controller data.

4. USER INTERACTION

4.1 Performance

When TC-11 first loads, the user is presented with the *Performance* view, which fills the entire iPad screen. The only interface object on screen is a button, tucked in a corner, which pulls out the view switching bar. Otherwise, the entire screen is waiting to be touched to activate synthesis.

Users often begin with a single touch when first engaging with TC-11. When they move their finger across the screen, they see graphic representations of the multi-point controllers in use, such as connecting lines, circles, and angle vertices. Eventually they discover the capability to use multiple touches, and further change the performance by moving those touches in different ways.

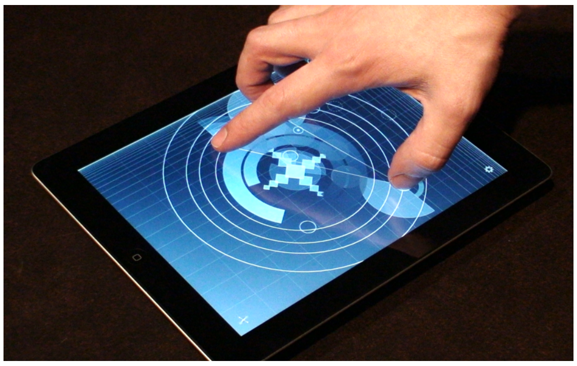


Figure 5: Performing TC-11 can use one or both hands, up to 11 touches.

Performance responsiveness is quick, although actual latency was measured to be around 40 ms. The majority of the latency comes from the touch screen. However, users report crisp responsiveness. This may be due to the fact that the capacitive touch screen activates with the initial contact with the finger, and not the pressure of the touch, which happens afterwards.

Teaching device motion performance to the user is a challenge. Device motion controllers rotate around one physical axis at a time. However, users often twist the device along multiple axes and simply listen for the result. The *Performance* view tries to inform performance by showing a series of sagging diamonds that fall along their prescribed axis. Even with this cue, some users have difficulty understanding the correct control motion.

The expressiveness of the instrument comes from the ability to drive many synthesis parameters with a variety of multi-point controllers, even with simple touch gestures. Basic multi-touch gestures can generate a complex set of controller responses, which translates to expressive synthesis results.

4.2 Patch Programming

The patch programming interface starts with a graph of the synthesis objects (Figure 2). Users tap on the objects to activate or deactivate them, and the view updates the patch connections to show the signal flow.

To access the synthesis parameters, users switch to the object group areas *(Oscillators, Filter / Amp, Effects)*. There they can map each synthesis parameter, using a standard visual interface (Figure 4). There is no graphical attempt to mimic an existing hardware layout, like the front plate of an analog synthesizer. The use of the standard interface instead of a virtual knob emphasizes the strong connection between parameters and their controllers.

New users who do not have experience with modular or analog synthesizers often lament the lack of easy note or scale generation. TC-11 uses its sequencer modules to accomplish this. However, incorporating the module as a frequency intermediary takes a working knowledge of the programming layout.

4.3 User Patches

TC-11 includes the ability to save custom user patches. These patches can be shared with other TC-11 users via email from within the app. Patches received from other users, when opened through the iPad's email client, load directly into TC-11.

5. REFLECTIONS

Following the release of TC-11, many users provided feedback about the performance capabilities, programming depth, and sound quality. These comments continue to inform the future of the instrument, and provide a reflection on the success or failure of design elements.

5.1 Enthusiasm

Provided below is an informal sample of comments from iPad App Store reviews and online iOS music blogs to show different responses to the instrument.

5.1.1 Interactivity and Expression

awkpod: "Even playing with just one patch has huge possibilities." (12/18/2011)
John Lehmkuhl: "I have seen many synthesizers with many different interfaces and love this new one you have where your fingers become the instrument." (2/5/2012)
David Shamban: "It's incredibly expressive and customizable. Just a word of warning, you may find yourself lost in a time hole playing with it for hours." (1/29/2012)

5.1.2 Patch Programming Depth

Subimage: "Still coming to grips with the routing, as it's quite complex - however it seems to generate the type of sounds you'd usually only get from things like Reaktor or Max/MSP." (12/16/2011)
Jfbbivdvj: "The patch parameters are virtually bottomless, affecting not only the sound itself but also the way it responds to your actions." (1/22/2012)
Tim Webb, www.discchord.com: "TC-11 is making my jaw drop every few minutes; realizing new and increasingly complex routing options." (12/18/2011)
Rich Courage: "Unconventional GUI that makes the best use of what the iPad is. You can get really deep into programming your own sounds." (2/6/2012)

5.1.3 Sound Production

KrisMcKenna : "The sound possibilities are amazing, and every time I pick it up something new happens." (1/22/2012)
valloy: "This synth looks great with endless possibility of shaping sound and its easy to navigate thru the settings." (1/28/2012)
David Isreal, www.smitematter.com: "A madhouse of sonic possibilities at your touch. A legitimate live performance, or studio (on the go) instrument that challenges and inspires." (1/14/2012)

5.2 Design Flaws

Some important features were not present in the initial release of TC-11. For example, the ability to internally record and export audio files to other iPad apps was not included. Many users considered the app incomplete without this capability. The feature was added in an updated version.

A big design failure was the lack of real-time audible feedback while editing a patch. The highly reactive, and sometimes unpredictable, responses from altering parameters was deemed difficult to predict. The iterative process of building a custom patch was slowed by constant switching between editing and performance. A live preview area was added to the *Patch* view to rectify the problem.

Another area for improvement is the graphical feedback given during performance. The aforementioned issue with drawing device motion shows the need for clearer graphical representations of controllers. Controllers that use time comparisons are also difficult to convey visually. More attention could be given to animating controllers to inform performance.

5.3 Future Work

While the design of TC-11 provides a deep programming experience, a clear improvement would be a setup that allows the user to design their own multi-point relationships.

A better implementation of the matrix patching system would allow for more than one controller to simultaneously control a single parameter. This could greatly increase parameter response potential.

Improved sharing of patches could be implemented using a patch repository: a server location for users to upload their own patches and browse other users' work. Sharing of actual performances is another option, using a system similar to the *World Listener* in Smule's *Ocarina* [6].

5.4 Concluding Remarks

TC-11 accomplishes many of its goals by providing an uncompromising combination of deep performance and customization. The multi-point controllers unlock the promised capabilities of the iPad's multi-touch screen, and deliver a fully expressive user interface. There is still much to learn from users who engage with it, both as a performance interface and programming environment.

6. REFERENCES

[1] P. Brinkmann, C. McCormick, P. Kirn, M. Roth, R. Lawler, and H.-C. Steiner. Embedding pure data with libpd. In *Proceeding of the Fourth International Pure Data Convention*, pages 291–301, June 2011.

[2] P. Dourish. *Where the action is: the foundations of embodied interaction.* MIT PRESS, 2011.

[3] G. Essl and M. Rohs. Interactivity for mobile music-making. *Organised Sound*, 14(2):197–107, 2009.

[4] G. Levin. *Painterly Interfaces for Audiovisual Performance.* Master Thesis, Massachusetts Institute of Technology, 2000.

[5] K. Schlei. Relationship-based implementation of multi-point data using a trackpad interface. In *Proceeding of the International Conference on New Interfaces for Musical Expression*, pages 136–139, November 2010.

[6] G. Wang. Designing smule‘s ocarina: The iphone‘s magic flute. In *Proceeding of the International Conference on New Interfaces for Musical Expression*, pages 303–307, November 2009.

[7] G. Wang, J. Oh, and T. Lieber. Designing for the ipad: Magic fiddle. In *Proceeding of the International Conference on New Interfaces for Musical Expression*, pages 197–202, November 2010.