

# TouchKeys: Software Manual

Version 0.2

Andrew McPherson  
Centre for Digital Music  
Queen Mary University of London  
<http://touchkeys.org>

September 7, 2015

## 1 Introduction

The TouchKeys are multi-touch sensors which attach to the surface of any piano-style keyboard. Each sensor measures the location and contact area of up to three fingers on the key surface. Touch data can be used with any MIDI synthesizer to add vibrato, pitch bends, timbre changes, and other expressive techniques.

The TouchKeys hardware connects to the computer by USB. This manual describes the software which operates the TouchKeys. The software combines data from the TouchKeys hardware with MIDI input from the underlying keyboard, generating a MIDI output that can be used to control any software or hardware synth.

The software may be downloaded from the SoundSoftware project site<sup>1</sup>. Full source code is also available.

### 1.1 System Requirements

The TouchKeys software runs on MacOS X, Windows and Linux.

#### 1.1.1 Mac OS X

- MacOS X 10.7 or later (tested up to 10.10). OS X 10.6 is supported up through August 2014 releases, but not in future releases.
- 32- or 64-bit system, Intel processor (any Mac since 2006)
- Two USB ports: one for keyboard, one for TouchKeys

---

<sup>1</sup><http://code.soundsoftware.ac.uk/projects/touchkeys>

#### 1.1.2 Linux

- 64-bit system, Intel processor. 32-bit build is possible from source.
- ALSA MIDI system. Use `a2jmidid` utility<sup>2</sup> to connect to the commonly used JACK MIDI environment.
- XWindows graphical environment
- Two USB ports: one for keyboard, one for TouchKeys

#### 1.1.3 Windows

- Windows Vista or later. XP support may be possible from source.
- 32-bit or 64-bit system.
- Drivers for TouchKeys device and MIDI loop-back. See <https://code.soundsoftware.ac.uk/documents/115> for details and current links.
- Two USB ports: one for keyboard, one for TouchKeys

## 2 Getting Started

Follow these steps to begin using the TouchKeys hardware and software. Refer to Figure 1 for more details on the software interface.

---

<sup>2</sup><http://home.gna.org/a2jmidid/>

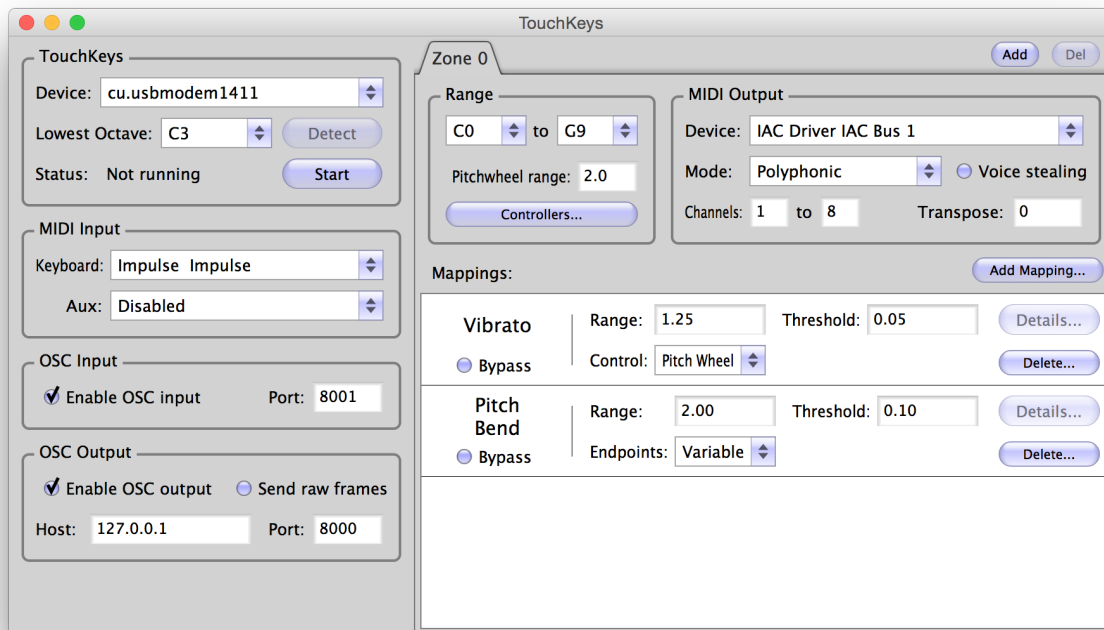


Figure 1: TouchKeys software control window.

## 2.1 TouchKeys Software Setup

1. Connect the TouchKeys to the computer via the mini-USB port.
2. Connect the MIDI keyboard to the computer via its USB port or a USB-MIDI converter. (Skip this step if using the TouchKeys in standalone mode, without keyboard.)
3. Launch the TouchKeys software.
4. Select the TouchKeys device in the upper-left corner of the control window (Figure 1). There is typically only one choice on most systems. If no devices appear, recheck the USB connections and select “Rescan Devices” from the Control menu.
5. In the MIDI Input box, select the keyboard on which the TouchKeys are installed. If using the TouchKeys without a MIDI keyboard, select “TouchKeys Standalone” in this box. In standalone mode, the touch rather than the key press will trigger the note.
6. In the TouchKeys box, click Start to begin running the TouchKeys device. Play a few notes, and the TouchKeys should automatically detect the octave of the MIDI keyboard.

7. Choose an output device in the MIDI Output box. This will be the port your synth software should listen to. If you select “Virtual Port” (Mac and Linux only), the TouchKeys software will create a virtual MIDI port other applications can listen to. On Mac, the IAC (Inter-Application Communication) MIDI Bus can also be helpful to connect applications to each other. This can be configured in Apple’s Audio MIDI Setup application. On Windows, select the port from the loopback MIDI driver you installed (e.g. LoopBe1 or loopMIDI).
8. Choose an output mode from the menu below the MIDI Output selection. For independent vibrato and pitch bends for each note, use Polyphonic mode. See Section 4.1.2 for more information.
9. Default mappings for vibrato and pitch bend are created for you. Change the mappings how you like, using the Add Mapping button to create new mappings. The startup behaviour can be changed in the application preferences (Section 6.4).

## 2.2 Synth Setup

You will need to configure your Digital Audio Workstation (DAW) to receive MIDI messages from the

TouchKeys. Each DAW is different, but the following general procedure applies:

1. The DAW should receive MIDI from the TouchKeys software. If a virtual port was used in the TouchKeys, look for a MIDI device called “TouchKeys”. Otherwise, select the appropriate IAC Bus (Mac), loop-back device (Windows) or ALSA/JACK MIDI Bus (Linux). **Important:** the DAW should *not* receive MIDI directly from the keyboard on which the TouchKeys are installed.
2. Create software instruments as needed. If using Polyphonic mode on the TouchKeys, an identically-configured software instrument will be needed on each MIDI channel used by the TouchKeys. This is needed because under the MIDI standard, pitch wheel and control change messages affect every note on a channel. If a different mode is used, then a single software instrument can be used on the MIDI channel indicated in the TouchKeys software.
3. For vibrato and pitch bends to operate as expected, pitch wheel should be enabled on the instrument. Check the pitch bend range on the instrument, and adjust it as necessary. If unsure of the pitch bend range, consult the instrument’s manual. Ranges are usually expressed in semitones (2 being the default on most instruments). You should enter this range in the “Pitchwheel Range” box in the TouchKeys software.

## 3 Device Selection

The software receives input from the TouchKeys USB hardware, and also takes input and output by MIDI and OSC<sup>3</sup>. A typical setup will use a TouchKeys device, one MIDI input device, and one or more MIDI output devices. OSC emulation can be used in place of the TouchKeys hardware<sup>4</sup>, and OSC output can be used to send touch data to other environments such as Max/MSP or Pd.

### 3.1 Input Devices

#### 3.1.1 TouchKeys

TouchKeys device selection is in the upper-left corner of the control window. The following controls are available:

- **Device:** Select a TouchKeys USB device.

<sup>3</sup><http://opensoundcontrol.org>

<sup>4</sup><https://code.soundsoftware.ac.uk/documents/>

- **Lowest Octave:** Set the lowest octave of the TouchKeys to match the MIDI keyboard. Best set automatically with the Detect button.
- **Detect:** Automatically detect the lowest octave of the TouchKeys. The default behaviour is to detect automatically when the Start button is pressed. This can be changed in the preferences (Section 6.4). Otherwise, to use it, make sure the TouchKeys are running and a MIDI input is selected. After pressing the button, the title should change to “Cancel”. Play a few keys and the button should return to saying “Detect”. The lowest octave will be updated according to the current setting of the keyboard. This will need to be updated whenever the transposition of the keyboard changes.
- **Start:** Start or stop running the TouchKeys device. Touch data is only collected while the device is running.
- **Status:** Shows the current status of the TouchKeys device. It should say “Running” when the TouchKeys are active.

#### 3.1.2 MIDI Input

This box is used to select the MIDI keyboard on which the TouchKeys are installed. If using the TouchKeys without a keyboard, or if using OSC emulation, select “TouchKeys Standalone” here.

A second box labelled “Aux.” is available for an optional second MIDI input device, for example an external control surface. MIDI data from this device will also be included in the output to the DAW.

#### 3.1.3 OSC Input

OSC input is used for two functions. First, it is used for TouchKeys emulation, in which an iOS device with the TouchOSC<sup>5</sup> app replicates the functions of the TouchKeys hardware. If using emulation, enable OSC input and select a port to receive messages on. This should match the settings of the TouchOSC app.

Second, OSC input can be used to control the functions of the GUI, for example changing mapping parameters or input/output devices. These messages all begin with the path `/touchkeys/control`. Messages for specific keyboard zones begin with `/touchkeys/control/segmentN` where N is the zone number (starting at 0).

<sup>5</sup><http://hexler.net/software/touchosc>

If not using either of these functions, OSC input can be left disabled.

## 3.2 Output Devices

### 3.2.1 MIDI Output

The keyboard can be divided into several zones, as described in Section 4. Each zone can have a separate MIDI output. In the default case (on startup), a single zone is used.

In the Device menu within the MIDI Output box, select the MIDI output port. On Mac and Linux, a virtual port can be used. This will create a software MIDI port that other applications can receive messages from. Otherwise, the software MIDI routing capabilities of the operating system can be used. On the Mac, the IAC Bus provides a means of routing MIDI messages from one program to another. Similar functionality is available on Windows via the third-party LoopBe and loopMIDI drivers. The IAC Bus can be enabled and configured in the Audio-MIDI Setup utility. On Linux, the JACK environment provides MIDI connectivity between programs.

Hardware MIDI ports can also be selected here to send TouchKeys messages to an external synth.

### 3.2.2 OSC Output

OSC messages can be sent from the TouchKeys software to other environments. Messages include notes played on the keyboard, touch locations, and outputs of various mappings (Section 5). The following settings are available:

- **Enable OSC output:** tick to enable sending OSC messages.
- **Send raw frames:** tick to enable sending raw TouchKeys data over OSC. This is useful for processing TouchKeys data in other environments such as Max/MSP, Pd and Supercollider. The bandwidth required for raw touch data can be significant, so leave this off unless needed.
- **Host:** the network address to send OSC messages to. `127.0.0.1` indicates the local machine.
- **Port:** the port to send OSC messages to. This should match the port that the receiving software listens on.

## 4 Keyboard Zones

The keyboard can be divided into multiple zones (ranges of notes), each with its own behaviour and mappings.

Each zone can also be sent to different MIDI devices. Multiple zones are useful for playing multiple instruments at once with the TouchKeys. They can also be useful when the TouchKeys are installed on only part of a keyboard, and you want to use the parts with and without sensors differently.

## 4.1 MIDI Output

MIDI output is also described in Section 3.2.1.

### 4.1.1 Device

This box selects the MIDI output device to use for this zone. Multiple zones can use the same MIDI device, or each can use a different device.

### 4.1.2 Output Mode

The output mode controls how MIDI channels are chosen for each note. The following options are available:

- **Passthrough:** this mode sends all MIDI messages to a single channel, without regard for how messages for one note will affect other notes. This mode is best used with polyphonic aftertouch messages, for synths that support such messages. Because MIDI pitch wheel and controllers affect all notes on a channel, playing multiple notes at once in Passthrough mode may have unpredictable results.
- **Monophonic:** this mode sends MIDI messages to a single channel, but only enables the mappings for the most recently played note. It is designed to be used with monophonic synths, for example emulations of orchestral instruments which play a single note at a time. Because only one note's mappings are active at once, control change and pitch wheel messages will not conflict with one another on the same channel. However, this mode is not suited for polyphonic instruments.
- **Polyphonic:** this mode allocates each note to its own MIDI channel. Pitch wheel and controller messages will be sent to that channel, allowing each note to be independently controlled in several dimensions. To use this mode, the synth must be configured to produce identical sounds on a range of MIDI channels. A **Voice Stealing** button indicates whether to silence old notes if no more MIDI channels are available.

### 4.1.3 Channels and Transposition

In Passthrough and Monophonic mode, a single MIDI channel can be selected. Choices are from 1 to 16. In Polyphonic mode, a range of MIDI channels can be indicated. Use caution when using Channel 10 as some synths assign it to drums, which can lead to unexpected results.

The Transpose box indicates the number of semitones to transpose the output for this zone. The default is 0, meaning no transposition. Transposing up an octave would be a value of 12; transposing down an octave would be a value of -12. Allowable range is four octaves (48 semitones) in either direction.

## 4.2 Zone Range and Settings

The range of each zone is adjustable. This affects which keys are part of the zone. It is generally assumed that the zones do not overlap in range; however this is not enforced, so it's possible for the same key to trigger outputs in multiple zones. Terminology: C4 is MIDI note 60.

### 4.2.1 Pitch Wheel Range

Under the MIDI standard, the range (amount of bend) of the pitch wheel is not defined. The Pitchwheel Range box sets the range of the pitch wheel in semitones. For example, a value of 2.0 means the maximum bend is 2 semitones (a whole step) up or down. The setting in this box is used to calculate pitch bends for all the mappings, so it should be set to match the pitch bend settings of the synth. Setting this box does not, by itself, affect any parameter of the synth; it only affects the scaling of the pitch wheel messages coming from the TouchKeys.

Some synths support a MIDI RPN message which sets the pitch wheel range. Using the Controllers button, the menu item "Send Pitchwheel Range RPN" will send a message to the synth on all active channels, telling it to update its pitch wheel range. Check whether your synth supports this RPN message before using it.

### 4.2.2 Controllers

Most MIDI keyboards have built-in controllers: pitch wheel, mod wheel, aftertouch, and often a collection of other controls. The Controllers button selects whether these incoming messages are resent to the MIDI output. In many cases, the TouchKeys will be used to send similar messages, so it can be useful to prevent the keyboard from sending these messages itself. These settings are adjustable for each zone. When Polyphonic mode is enabled, any enabled controller messages are broadcast to all active channels.

Available options are Pitch Wheel, Mod Wheel (CC 1), Aftertouch and Other Controllers. All MIDI controllers except CC 1 fall under the Other Controllers category.

## 4.3 Adding and Removing Zones

Keyboard zones can be added and removed using the buttons at the top right of the window. Up to 8 zones can be active at a time.

# 5 Mappings

A *mapping* is a relationship between touch and sound. Mappings take the touch and MIDI data from the keyboard and use it to generate a MIDI or OSC output stream for controlling a synth.

Mappings in the TouchKeys software are modular: each mapping provides one specific behaviour, and multiple mappings can be loaded at once. Mappings are local to each zone of the keyboard.

## 5.1 Adding and Deleting Mappings

When the TouchKeys software launches, no mappings are loaded. Use the "Add Mapping" button on the right-hand side of the window to create a new mapping. A menu appears with a choice of several types of mapping, as described below.

Once a mapping has been created, it appears in the list, where its parameters can be changed. Click the Delete button to remove the mapping from the keyboard zone.

## 5.2 Bypassing Mappings

Each mapping can be temporarily bypassed without being deleted by ticking the Bypass box on the left-hand side of the list.

## 5.3 Vibrato

The vibrato mapping uses the side-to-side (X) motion of the finger to create a vibrato effect. It can be configured to use a pitch-based vibrato (the default) or to modulate a controller value for amplitude-based vibrato (useful on some wind instruments, for example).

### 5.3.1 Vibrato Parameters

The vibrato mapping has several configurable parameters:

- **Range:** This sets the maximum width of the vibrato in semitones (for the default pitch wheel vibrato). The actual width of the vibrato, as well as its speed, depends on the size and rate of the hand motion. If a control other than pitch wheel is selected, this sets the maximum deviation of the controller value. *Typical values for pitch wheel: 0.2-1.5; typical values for other controllers: 5.0-50.0.*
- **Threshold:** The vibrato mapping waits until it detects a back-and-forth motion to engage. This prevents the notes from drifting out of tune unintentionally. The threshold parameter sets the size of the motion that must be detected before vibrato engages. *Typical values: 0.02-0.2.*
- **Control:** This selects between pitch wheel vibrato (the default) and a different controller. Any MIDI controller can be selected. If a different controller is selected, the value from the vibrato mapping is added on top of the incoming controller value from the keyboard. *Typical value: Pitch Wheel.*
- **Range:** The maximum range of the pitch bend, in semitones. The actual amount of pitch bend at any given time depends on how far the finger moves along the key. *Typical values: 1.0-7.0, with 2.0 a common choice.*
- **Threshold:** The pitch bend mapping waits for the finger to move a minimum distance before beginning to change the pitch. This prevents notes from bending unintentionally. The larger the threshold, the larger the motion required to engage the pitch bend. *Typical values: 0.05-0.2.*
- **Endpoints (Fixed):** There are two choices for Endpoints, Variable and Fixed. In Fixed mode, moving the finger to the end of the key always produces the amount of bend specified in the Range box, regardless of where the finger began. For example, with a range of 2 semitones and Fixed endpoints, pressing the note halfway along the key and moving the finger to the end will bend the note 2 semitones. Pressing the finger three-quarters of the way along the key and moving to the end will also bend by 2 semitones, even though the distance travelled by the finger is smaller.
- **Endpoints (Variable):** In Variable mode, the amount of pitch bend is proportional to the distance travelled by the finger. In this mode, with a range of 2 semitones, pressing the finger one-quarter of the way along the key and moving to the nearer end would bend the note by 0.5 semitones (one quarter of the total range); moving the finger to the farther end would bend by 1.5 semitones.

### 5.3.2 Vibrato Performance

The vibrato mapping requires that your software instrument or synth support pitch bend capability (unless a different controller is selected). Also make sure the pitch wheel range on the synth matches the value in the Range box (Section 4.2.1).

To play the vibrato mapping, rock the hand back and forth on the key while the note is held down. You should see the touch move from side to side on the keyboard display (Section 6.1). **Note:** the narrow parts of the white keys (between the black keys) do not support X-axis position measurement, so the vibrato mapping cannot be used in this region of the key. However, you can add vibrato to a note originally played in this area by placing a second finger on the wide part of the white key and performing vibrato with this finger.

## 5.4 Pitch Bend

The pitch bend mapping uses motion on the long (Y) axis of the key to bend the pitch up and down. Moving the finger upward (away from the player) bends the pitch up; moving the finger toward the player bends the pitch down.

### 5.4.1 Pitch Bend Parameters

The pitch bend mapping has the following configurable parameters:

### 5.4.2 Pitch Bend Performance

The pitch bend mapping requires that your software instrument or synth support pitch bend capability. Also make sure the pitch wheel range on the synth matches the value in the Range box (Section 4.2.1).

To play the pitch bend mapping, move the finger up and down on the key after the note is played. (No matter where the touch begins, the pitch will be what is normally expected from playing that key; it is the motion and not the initial location that determines the pitch.) You should see the touch move up and down on the keyboard display (Section 6.1).

## 5.5 Control

The control mapping can assign any touch dimension to any MIDI controller, including aftertouch and pitch

wheel. It is quite versatile, but not every setting will be equally easy to play. Multiple control mappings can be used at the same time with different settings.

### 5.5.1 Control Mapping Parameters

The control mapping has the following parameters:

- **Parameter:** This selects the touch parameter to use. Possible values are *X Position* (side-to-side), *Y Position* (long axis of the key), *Contact Area* (size of the touch, e.g. from playing with fingertip or pad of finger), *2-Finger Mean* (average Y position when two fingers are used on one key) and *2-Finger Distance* (difference in Y position when two fingers are used on one key).
- **Input Range:** The Position and Contact Area are scaled between 0 and 1 on each key, where 0 is bottom/left, and 1 is top/right. The Input Range parameter selects how to scale this range to produce a MIDI output. For example, if using Y position, settings of 0.0 to 1.0 would map the entire length of the key to the entire range of the MIDI controller. Settings of 0.0 to 0.5 would map only the lower half of the key to the entire range of the MIDI controller; touches in the upper half ( $Y > 0.5$ ) would produce the maximum value on the controller. *Typical values: 0.0-1.0 when using Absolute type; often a smaller range when using Relative types (see below).*
- **Control:** Selects which MIDI controller to use for an output. In addition to the standard controllers, pitch wheel, channel pressure (mono aftertouch) and polyphonic aftertouch can be used.
- **Type:** The three choices are *Absolute*, *1st Touch Relative* and *Note Onset Relative*. In Absolute mode, the current value of the position or contact area is used directly to calculate the MIDI controller value. In 1st Touch Relative mode, the difference between the initial position/contact area and the current value is used. In Note Onset Relative mode, the difference between its value at the start of the note and the current value is used. The finger often contacts the key well before the key is pressed, so 1st Touch Relative and Note Onset Relative types will produce subtly different results. *Typical Values: Absolute or Note Onset Relative.*

### 5.5.2 Control Mapping Detailed Parameters

Clicking the “Details” button next to the Control mapping will bring up a window with further adjustable param-

eters. These extra parameters include:

- **Output Range:** Selects the range of MIDI controller values to send. For example, setting the output range to 0-64 will scale the entire range of the input to produce control values between 0 and 64. *Typical values: 0-127 for most controls; 0-16383 for pitch wheel.*
- **Default Output:** Selects the MIDI controller value to send at the beginning of the note, before any finger movements have taken place. This might need to be changed to match one end of the output range. *Typical value: 0.*
- **Threshold:** This is only used for *1st Touch Relative* and *Note Onset Relative* types. Specifies the minimum change in value that must take place before the output begins to change. *Typical values: 0-0.1 for position-based mappings; 0.1-0.3 for contact area mappings.*
- **Direction:** This is only used for *1st Touch Relative* and *Note Onset Relative* types. The *Normal* setting maps increasing input values to increasing output values, and is the default behaviour. The *Reverse* setting maps decreasing input values to increasing output values (e.g. moving the finger downward could increase the output). The *Always Positive* setting maps a change in either direction to increase output values. *Typical value: Normal.*
- **Out of Range:** Selects the output behaviour when the input goes outside of the range specified in Input Range. Values are *Ignore* (send no output when the input is out of range), *Clip* (send the min or max output when the input is out of range) and *Extrapolate* (send a scaled output which may be outside the values specified in Output Range). *Typical value: Clip.*
- **Use 14-bit CC:** Selects whether the given MIDI controller should have 14-bit resolution. 14-bit CCs are created by using pairs of MIDI controllers together. The MIDI specification supports 14-bit resolution on controllers 0-31, where controllers 32-63 represent the lower 7 bits. The TouchKeys software allows 14-bit support to be selected on any controller except Channel Pressure and Poly Aftertouch. If your synth does not support 14-bit CCs, the results may be unexpected. Pitch Wheel is always 14 bits.
- **Ignore 2/3 Fingers:** Disables output from the mapping when 2 or 3 fingers are present on the key. This

is useful for having multiple Control mappings active simultaneously, one of which is using 2-Finger Mean or 2-Finger Distance, by making sure that only one mapping at a time responds. *Typical values: disabled.*

### 5.5.3 Control Mapping Performance

To use the control mapping, ensure that the synth responds to the MIDI controller you have selected in the Control box. How it is played depends on the Parameter and Type settings; for example, the current position of the finger might be used to modulate a filter cutoff, or the relative change in contact area could be used to change the volume of a note.

The control mapping can be used to send polyphonic aftertouch messages. Polyphonic aftertouch is specific to each note, and it allows note-by-note control of an instrument using a single MIDI channel. For all other controllers, independent control of multiple notes at once requires the use of Polyphonic mode (Section 4.1.2) where each note gets its own MIDI channel.

**Note:** sending certain messages, such as mod wheel (CC1) or channel pressure (mono aftertouch) might conflict with messages sent from the keyboard itself if the keyboard has those controllers. Generally speaking, where a controller value is received from the keyboard and another one is generated by the TouchKeys mappings, the two are added together at the output. It may also be useful in certain situations to disable the keyboard controllers; see Section 4.2.2 for details.

## 5.6 Multi-Finger Trigger

This mapping causes a specific action to take place whenever the user taps the key with two or three fingers. It can be used, for instance, to retrigger the MIDI note without releasing the key, which is useful for playing fast tremolos.

### 5.6.1 Multi-Finger Trigger Mapping Parameters

- **Touches:** How many simultaneous fingers on the key are needed to activate the trigger. *Typical value: 2.*
- **Repeat Taps:** How many times the fingers have to tap before the trigger activates. A value of 1 will activate the trigger as soon as the specified number of touches are present on the key. Higher values reduce the likelihood of accidental triggering. *Typical value: 1.*

- **Note:** Which MIDI note to send when the trigger activates. It can be the same as the note played, or it can be a fixed note value, useful for activating keyswitches. *Typical value: Same.*
- **Also Send on Release:** Enabling this option will cause the note to be sent again when the fingers are released from the key after the trigger is sent. Enabling it is useful for even faster repeated notes, for example simulating tremolos on a violin. *Typical value: disabled.*

## 5.7 Split Key

This mapping divides the 12 keys of the keyboard into 24 or more notes by triggering a different pitch in different regions of the keys. It is designed for performing microtonal music. **Note:** accurate microtonal tuning requires that the Pitchwheel Range be set to match the synth (Section 4.2.1), and that the synth has a linear response to the pitch wheel.

### 5.7.1 Split Key Mapping Parameters

- **Tuning:** Selects a microtonal tuning. Options include 19, 24, 31 and 36-tone equal temperament and the Yarman-24c Turkish maqam scale. 19ET, 24ET and Yarman-24c divide each key in two segments. 31ET and 36ET divide each key in three segments. For example, 24ET will make the back half of each key a quarter-tone higher than the front half.
- **Control:** Controls whether the microtonal tuning is selected by finger location, number of touches or both. If “Both” is selected, then if the key is divided into two segments, the second segment will be chosen if either the touch is on the back half of the key, or two fingers are down (and a similar idea for three segment divisions). In some cases one technique will be more ergonomic than the other. *Typical value: Both.*
- **Retriggerable:** If this option is selected, then adding a new touch in a different segment of the key will cause the note to retrigger with a new tuning. It is useful for playing passages where microtones on the same key follow one another in quick succession.

## 5.8 Release Angle

This mapping measures the speed at which finger moves along the key when the MIDI note ends. It can be used to add buzzes, falls and other expressive ornaments at the



ends of notes by triggering keyswitches supported by the synth.

All the settings for the Release Angle mapping can be reached from the Details button.

### 5.8.1 Release Angle Mapping Parameters

- **Preset:** This box populates the rest of the fields with values tuned for particular synths. Currently supported synths are Sample Modeling Trombone and Sample Modeling Trumpet.
- **Window Length:** This mapping calculates the speed at which the finger is moving when the MIDI note is released. The window length sets the period over which the speed calculation is based. Shorter values will be more reactive to short gestures at the end of notes, but more sensitive to false triggers. The default value is 100ms.
- **Release Moving Up:** Playing “release moving up” means to push the finger forward along the key (i.e. away from the body) just as the key is being released. This gesture can be used to trigger keyswitches. The *Min. Release Speed* sets a threshold for how fast the finger needs to be moving before the effect is activated. If the speed exceeds the threshold, then up to three MIDI notes are sent right before the note is released. The note numbers and velocities of these notes are entered in the box below. Entering 0 for both values means the note will not be sent.
- **Release Moving Down:** Playing “release moving down” means to pulling the finger along the key toward the body just as the key is being released. Its settings are analogous to “release moving up.”

## 5.9 Onset Angle

This mapping is still experimental as it is not yet polished. It can be enabled by selecting “Enable Experimental Mappings” from the Control menu (menu bar on the Mac, or top of the window on Linux/Windows).

This mapping measures the speed of the finger when the MIDI note begins. It can be used to add scoops and other expressive ornaments at the beginnings of notes. However, it does not yet have any editable settings so its behaviour needs to be adjusted in the code.

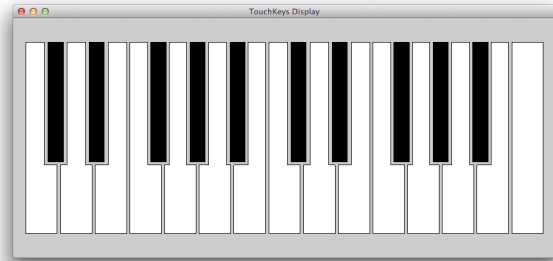


Figure 2: TouchKeys keyboard display.

## 6 Other Features

### 6.1 Keyboard Display

The keyboard display can be found in a separate window. It shows the current status of the TouchKeys and MIDI data. Keys pressed on the MIDI keyboard will highlight in red. Touches will appear as purple or green circles, for the white and black keys, respectively, with the size of the circle proportional to the contact area of the touch.

The window can be brought to the front by selecting “Keyboard Display” from the Window menu.

### 6.2 Data Logging

Raw data from the TouchKeys sensors and from the MIDI keyboard can be logged to a file for later analysis. This is selected from the Control menu. Playback of log files is currently supported at a very basic level, but with few extra features. Log files are created for touch and MIDI data in the current user’s Desktop folder.

This feature is intended mainly for analysis and research, though in future versions it could be expanded to allow the same performance to be replayed with different mappings.

### 6.3 Saving and Loading Presets

Presets can be saved and loaded from the File menu. A preset contains information on keyboard zones and mappings within each zone. Device selections are deliberately not saved with presets so that they can be interoperable between different machines.

Also in the File menu, the “Clear Settings” option removes all mappings and returns the setup to a single keyboard zone.

## 6.4 Preferences

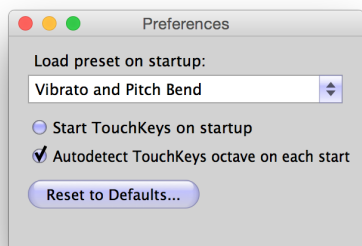


Figure 3: TouchKeys preferences window.

Application preferences can be accessed from the Control menu. The following settings can be changed:

- **Load preset on startup:** Controls which settings are loaded when the program first starts. The default setting, “Vibrato and Pitch Bend,” is a useful starting point which works with most synths. Other options are “None” (no mappings), “Last Saved” (loads the last saved preset), and “Choose” (select a specific file to be loaded at startup).
- **Start TouchKeys on startup:** If enabled, the TouchKeys device will automatically begin running when the program launches.
- **Autodetect TouchKeys octave:** If enabled, then whenever the TouchKeys device starts, the octave detect function is enabled (Section 3.1.1). This will automatically assign the TouchKeys sensors to the correct octave of the keyboard.

## 7 Troubleshooting and Support

Here are a few common issues and possible solutions:

- *The TouchKeys device does not appear in the list.* Ensure that the mini-USB cable is properly attached to the USB port on the TouchKeys. Depending on how the instrument is built, it may be a tight fit. Be sure that the TouchKeys were attached to the computer before launching the software.
- *The MIDI keyboard does not appear in the list.* Check the USB cable or the USB-MIDI converter. On Mac, launch Audio MIDI Setup. Select “Show
- *When I click Start, the TouchKeys status says “Failed to Open”.* Check that the TouchKeys device is still connected, and that the device you have selected is an actual TouchKeys device. Other USB-Serial devices may appear in this list. Check that you have permissions on your system to access USB devices.
- *I don’t get any MIDI output when I play.* Check that you see touch data in the Keyboard Display window, and that the keys highlight in red when you play them. If not, check that the TouchKeys are running and that you have selected the right MIDI Input. If the display is correct, check that you have selected a MIDI Output port and that your synth is listening on the same port. Make sure the TouchKeys and your synth are using the same MIDI channel(s).
- *I get a double output when I play, or the notes sound strange.* Check that your synth is listening *only* to the TouchKeys MIDI output and not directly to the MIDI keyboard.
- *I get an output, but there is no vibrato, pitch bend, etc.* Check that your synth is listening to the TouchKeys software and not directly to the MIDI keyboard. Check that you have added mappings for the TouchKeys. Make sure the TouchKeys are running and that the touch data appears in the Keyboard Display window. Also make sure the touch data and the MIDI keyboard are aligned (the touch circles should line up with the red highlighted keys). If they are not aligned, click the Detect button in the upper-left section of the main control window to realign the octaves.
- *Whenever I play vibrato and pitch bend, all the notes bend together.* Make sure you are using Polyphonic mode and that you have created several copies of your software instrument, each listening on a different channel.
- *The pitch bend is out of tune or unexpectedly wide/narrow.* Make sure the Pitchwheel Range (top centre of control window) matches the settings on your synth.

MIDI Window” from the Window menu and click the “Rescan MIDI” button. If this is unsuccessful, check if your keyboard requires a driver. On Windows, make sure the keyboard drivers are installed. On Linux, ensure that your MIDI keyboard is visible to the ALSA MIDI system.

- *The notes start immediately when I touch the key, not when I press it down.* Make sure you have selected your MIDI keyboard in the MIDI Input box, not TouchKeys Standalone Mode.
- *The TouchKeys occasionally freeze while I play.* This is a known bug with certain USB hubs, though the exact cause is still unknown. It is recommended to attach the TouchKeys directly to the computer rather than using a hub.

## 7.1 Further Information

For further information and assistance, please join the TouchKeys forum<sup>6</sup>.

## 8 OSC Control Messages

The TouchKeys software can be controlled from an external source via OSC messages. Ensure that OSC Input is switched on and that your external software is sending to the port listed in the OSC Input box.

Unless otherwise noted, each received message will send back an OSC message with the result of the action. This message will have path `/touchkeys/control/result`, type `i` and a value of 0 or 1 depending on whether the operation succeeded or failed. To receive these messages, OSC Output must be enabled and the host and port must be set to match the external device/software from which you are controlling the TouchKeys.

---

<sup>6</sup><https://groups.google.com/d/forum/touchkeys>

OSC Path (beginning with /touchkeys/control/)	Types	Data Contents	Description
preset-load	s	Full path to preset file	Loads a previously saved preset.
preset-save	s	Full path to preset file	Saves the current preset to file.
preset-clear	-		Clears the current setup to one zone, no mappings.
tk-list-devices	-		Returns a list of current TouchKeys devices in OSC message /touchkeys/control/tk-list-devices/result, type i... where i is number of devices, and ... is one or more paths of type s
tk-start	s	Device path	Start data gathering from TouchKeys device
tk-stop	-		Stop data gathering from TouchKeys devices
tk-set-lowest-midi-note	i	Lowest MIDI note	Set lowest MIDI note of TouchKeys device. Should be a C.
tk-autodetect	-		Start autodetecting TouchKeys-keyboard alignment
tk-autodetect-stop	-		Stop autodetecting TouchKeys-keyboard alignment
list-midi-in	-		Returns a list of current MIDI input devices in OSC message /touchkeys/control/list-midi-in/result, type i... where i is number of devices, and ... is one or more pairs of type i s which indicate device number and name, respectively.
list-midi-out	-		Returns a list of current MIDI output devices (see above)
set-midi-in-keyboard	i or s	Number/name of device	Sets the MIDI keyboard device. If value begins with "stand" (standalone), TouchKeys standalone mode is enabled. If a negative integer is passed, MIDI keyboard input is disabled.
set-midi-in-aux	i or s	Number/name of device	Sets the MIDI aux device. If a negative integer is passed, MIDI aux input is disabled.
add-segment	-		Adds a new keyboard zone (segment).
delete-segment	i	Segment number	Deletes the specified keyboard zone (segment).
segmentX/set-midi-out	i or s	Number/name of device	Sets the MIDI output device.
segmentX/list-mappings	-		Returns a list of mappings in OSC message /touchkeys/control/list-mappings/result, type i... where i is the # of mappings, followed by one or more names (type s)
segmentX/add-mapping	i or is	Mapping type, Optional OSC path	Creates a new mapping. Types 0-6 are Control, Vibrato Pitch Bend, Split Key, Multi-Finger Trigger, Onset Angle, Release Angle
segmentX/set-range	ii	Lower/upper MIDI notes	Sets range of keyboard zone
segmentX/set-transpose	i	Semitone transposition	Sets output transposition of keyboard zone.
segmentX/set-transpose-octave-up	-		Increases transposition by 12 semitones, up to 48 max.
segmentX/set-transpose-octave-down	-		Decreases transposition by 12 semitones, down to -48 min.
segmentX/set-controller-pass	iiii	Channel pressure, pitch wheel, mod wheel (CC 1), other CCs	Sets whether to pass (1) or block (0) various messages from keyboard.
segmentX/set-pitchwheel-range	i or f	Semitone range	Sets the pitchwheel range for zone.
segmentX/send-pitchwheel-range	-		Sends a MIDI RPN to the output which sets pitchwheel range of the synth.
segmentX/set-midi-mode	s	Mode	Sets mode to passthrough (pass), monophonic (mono) polyphonic (poly) or off (off).
segmentX/set-midi-channels	ii	Lower/upper channels	Sets the channel range of the keyboard zone. Only used for mono (first argument only) and poly.
segmentX/set-midi-stealing	i	Enable stealing	In polyphonic mode, if set to 1, voice stealing enabled.

Table 1: List of OSC control messages supported by the TouchKeys software. All messages begin with /touchkeys/control/. For paths including segmentX/, X should be replaced with the segment (zone) number. Types i, f and s denote integer, float and string, respectively.

OSC Path	Types	Data Contents	Description
<b>All Mapping Types</b>			
set-bypass	i	Bypass?	If argument is nonzero, bypass this mapping.
set-active-notes	i	Bits 0-11: active notes	Each bit represents one pitch class (0 = C, 1 = C# etc.). Set a bit to 1 to enable the mapping on that pitch class. 0xFFFF (4095) is all notes.
<b>Control</b>			
set-input-parameter	i	Input parameter	Values 1-5 correspond to X position, Y position, size, 2-finger mean, 2-finger distance.
set-input-type	i	Input type	Values 1-3 correspond to Absolute, 1st-Touch Relative, Note-Onset Relative.
set-input-range-min	f	Input value	Sets lower bound of the input range.
set-input-range-max	f	Input value	Sets upper bound of the input range.
set-input-range-center	f	Input value	Sets centre of the input range.
set-output-range-min	f	Output value	Sets lower bound of the output range.
set-output-range-max	f	Output value	Sets upper bound of the output range.
set-output-default	f	Output value	Sets default output at start of note.
set-out-of-range-behavior	i	Behavior type	Values 1-3 correspond to Ignore, Clip, Extrapolate.
set-midi-controller	i	MIDI controller	Which controller to map to, or 128 = Pitch wheel, 129 = Channel pressure, 130 = Poly aftertouch.
set-threshold	f	Threshold	Only for relative types. Min. change before mapping activates.
set-ignores-multiple-fingers	ii	2/3 fingers	If each value is 1, ignore 2 or 3 finger frames, respectively.
set-direction	i or s	Direction	Only for relative types. Values 1-3 correspond to Normal, Reverse, Always Positive.
<b>Vibrato</b>			
set-vibrato-control	i	MIDI controller	Which controller to map to, or 128 = Pitch wheel, 129 = Channel pressure, 130 = Poly aftertouch.
set-vibrato-range	f	Semitones	Set range of vibrato output in semitones.
set-vibrato-prescaler	f	Prescaler	Sets the prescaler value before a nonlinear tanf() function. Internal value which rarely needs changing. Default is 2.0.
set-vibrato-timeout	f	Timeout (sec.)	Set how much inactivity before vibrato deactivates.
set-vibrato-threshold	f	Threshold	Set amount of bidirectional motion before vibrato activates.
<b>Pitch Bend</b>			
set-bend-range	f	Semitones	Set max pitch bend in semitones.
set-bend-threshold	f	Threshold	Set minimum motion before pitch bend activates.
set-bend-fixed-endpoints	f	Buffer at end	Set the bend type to fixed endpoints, specifying how far from each end of the key the endpoints are located (0 = end of key).
set-bend-variable-endpoints	-		Set the bend type to variable endpoints.
set-bend-ignores-multiple-fingers	ii	2/3 fingers	If each value is 1, ignore 2 or 3 finger frames, respectively.
<b>Split Key</b>			OSC control not yet implemented.
<b>Multi-Finger Trigger</b>			OSC control not yet implemented.
<b>Onset Angle</b>			OSC control not yet implemented.
<b>Release Angle</b>			OSC control not yet implemented.

Table 2: List of OSC control messages for individual mappings. These should be prefixed with /touchkeys/control/segmentX/mappingName/ where X is the zone number and mappingName is the name of the mapping, as returned by the list-mappings message.