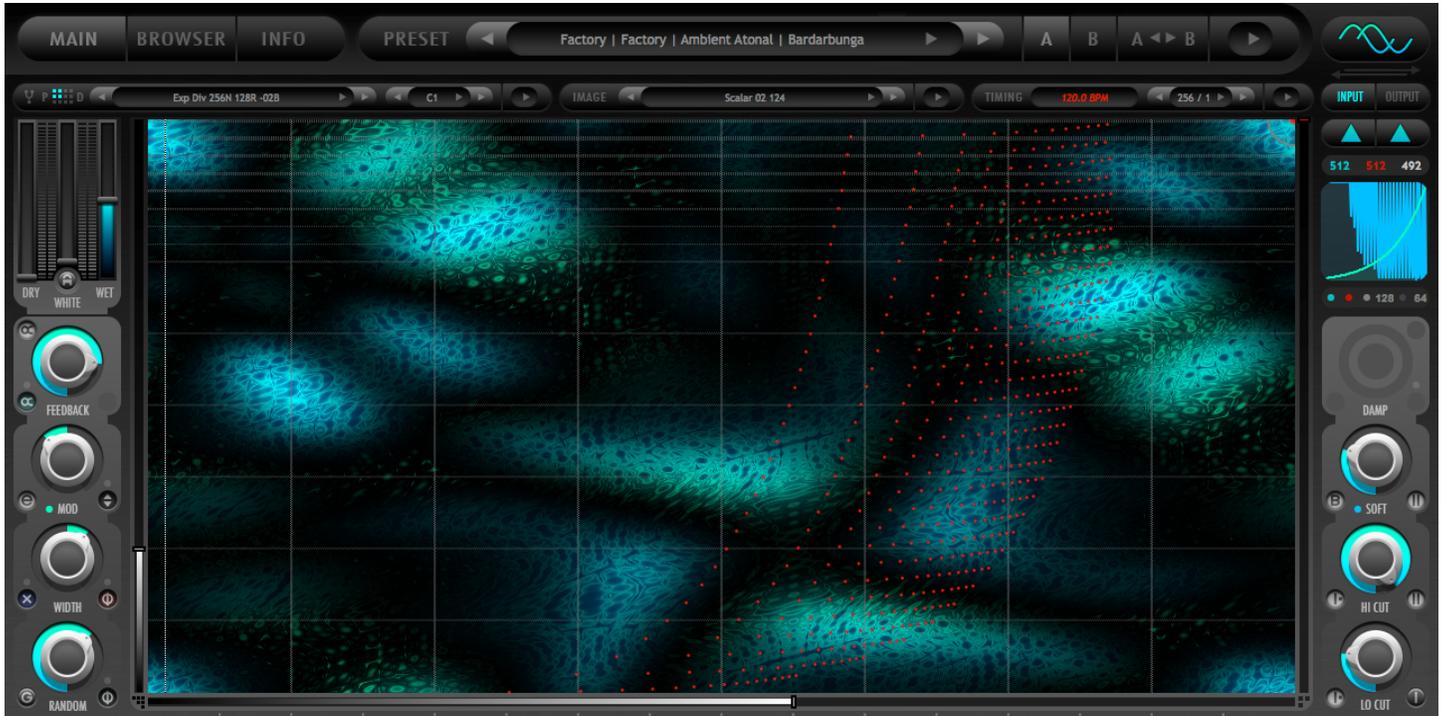


Kaleidoscope

2audio

DENIS MALYGIN
ANDREW SOUTER



Credits

- **Concept, Algorithm Mathematics, Functionality, & Feature Development:** Andrew Souter
- **Systems & GUI Coding, SDK Infrastructure, DSP Optimization & Multi-threaded Engine:** Denis Malygin
- **GUI Design:** Andrew Souter
- **Image, Tuning Scale, and Waveform Resource Library:** Andrew Souter
- **Presets:** Andrew Souter, Denis Malygin, Simon Stockhausen, Sascha Dikiciyan, Jim Hurley, Duncan Farmer, Austin Noble
- **Manual:** Andrew Souter
- **Additional Helpful Insights:** Teemu Voipio

Thanks

Many thanks to our beta team for helping us release bug-free software. Additional thanks to all of the great minds who have come before us as well as our contemporary peers who continue to help advance the art and science of digital audio signal processing. And finally, special thanks to our friends and families who are kind and forgiving enough to tolerate the level of obsessiveness necessary to make a product such as Kaleidoscope.

Table of Contents

1) Introduction

- a. Welcome to 2CAudio
- b. Welcome to Kaleidoscope
- c. System requirements
- d. Installation & Authorization
- e. File Management
- f. Uninstalling

2) The GUI

- a. The Navigation Bar
- b. The Main Page
- c. The Browser Page
- d. The Info Page
- e. Mouse and Keyboard Interface Gestures

3) Getting Started

- a. So what is Kaleidoscope?
- b. What is a resonator?
- c. What is an Image Map?
- d. Why is Tuning Important?

4) Main Page Parameters & GUI Details

- a. Parameter Overview
- b. Displays
- c. Knobs
- d. Mix Controls & Sliders

5) Browser Page Parameters & GUI Details

- a. Parameter Overview
- b. The Preset Browser

6) Info Page Details

- a. Parameter Overview
- b. Preferences
- c. Karma Boost Area

7) Appendix

- a. File Formats
- b. Parameter Appendix

Introduction

Welcome to 2CAudio

First, we would like to thank you for your purchase and welcome you to the 2CAudio family. 2CAudio develops industry-leading audio plug-ins in VST, AU, and AAX formats. Its primary focus is on spatial processing, advanced creative effects, and other future-forward ideas. The 2CAudio corporate motto is: **Convergent Creative Precision**.

Convergent represents our understanding of the universal trend in knowledge and discovery that tells us that over time disparate segments of art, science, technology, and thought merge and become one, and therefore wisdom and progress in one discipline can often be successfully applied to another.

Creative represents our commitment to innovation, independent thought, and invention. We learn from the past, but we do not wish to limit ourselves to mere emulation and duplication. We seek not to emulate the great minds that have come before us, but rather to understand the questions they struggled with, and explore new solutions, ideas, and discoveries using the latest tools of our generation. Moreover, our products are designed to encourage these ideals in our users, and our largest reward is hearing of enhanced creativity that was inspired by our products.

Precision represents our obsessive commitment to rigorous scientific standards of technical and numerical accuracy. The words “Good Enough” are simply not in our vocabulary, and we strive for perfection in all areas of our endeavors.

2CAudio is passionate about both art and science and we consider them both invaluable leaves on the tree of knowledge.

2CAudio prides itself on active engagement with our customers and the market as a whole, and we are always open to your feedback. Our corporate vision is not immutable, and we attempt to mold and form our products to the needs of our customers. We value your input and thank you for providing it to us. We are always listening at info@2caudio.com.

Technical support is provided via email at support@2caudio.com.

To learn more about 2CAudio, please visit our website at www.2caudio.com.

Welcome to Kaleidoscope

Congratulations on your purchase of Kaleidoscope. We hope you enjoy the journey. It really is quite a trip! Kaleidoscope is an entirely new class of effects processors and is one of the most unique signal processing effects to come to market in recent history. At the risk of sounding immodest, it may very well become one of the defining sounds of 21st century. Kaleidoscope is a massively parallel bank of physically modeled resonators that can be tuned completely arbitrarily with scientific precision and are dynamically modulated over time via over two million points of automation in the form of two independent image maps. In simplistic terms, Kaleidoscope uses pictures to control sound!

Kaleidoscope's primary uses are:

- Custom Sound Design
- Dynamic Filter Effects
- Dynamic Resonator Effects
- Morphing Delay Effects
- Special Effects Processing & Generation
- Extreme Nano Textural Effects
- Vocoder-like effects
- Extreme Spatialization and FIRs
- Algorithmic Composition & Inspiration
- Rhythmic Filtering and Synth Patterns
- Harmonic & Tonal “reverb”
- Ambient Music Generation

System requirements

Minimum Recommended Hardware: 4-core CPU with SSE2 support running at least at 2.0GHz and 4GB of RAM.

Kaleidoscope is a massively parallel algorithm with dynamic CPU usage. CPU usage scales linearly depending on the number of enabled lines (voices) used in a particular preset. The minimum number of active lines is 1 and the maximum is 512. **Using 512 lines requires roughly 512 times as much CPU power as using 1 line!** Additional factors such as the current Oversampling Ratio, Resonator Mode, Modulation Mode, and others also affect CPU usage.

Kaleidoscope is multi-threaded and can distribute its CPU load across a variable number of threads and cores depending on the current setting in the Info Page Preferences. It can be set to use all available cores on your system including virtual logical cores from hyper-threading. Kaleidoscope offers extreme control over threading behavior to allow you to find the best match for your current hardware, OS, host, and project needs.

Kaleidoscope has been optimized for two things: extreme Real-Time performance relative to the amount of computational complexity it represents, and extremely fast Offline render times even when using the most extreme settings. This is achieved both by multi-threading as well as offering a variable Buffer Size preference that allows users to choose between maximum efficiency and lowest latency.

Using the maximum number of lines together with Oversampling at low Buffer Size settings can require a large amount of CPU resources. Oversampling offers only a very minor benefit to the Spring Resonator Modes, and no benefit at all to the FIR Resonator mode; it offers the most benefit to the String Resonator Modes. We therefore recommend simply running at 1X Oversampling for Real-Time use, and reserving Oversampling for Offline bounces.

Kaleidoscope is a very different type of tool compared to 2CAudio reverb products. It is generally assumed that reverb products will be used live on track inserts or auxiliary busses and potentially many instances will be kept continuously running in a host session for many hours in the working day. Tools such as reverb effect mix decisions and therefore are typically kept live in sessions. Kaleidoscope is designed more as a sound-design and special FX tool. It should be thought of as content creator and extreme FX processor. While it is possible to run multiple instances live in projects, particularly when using large Buffer Size settings, the recommended work flow is to bounce or freeze KS instances once a desirable result has been achieved. Running many instances of Kaleidoscope live with maximum settings will quickly consume all available resources even on the fastest computers available in 2014-2015. It is important to get acquainted with the process of rendering audio FX in your host application via bouncing, freezing, or rendering-in-place to get the most out of Kaleidoscope. **This is particularly true when using a low Buffer Size preference. Using the maximum Buffer Size setting VERY significantly increases efficiency, and dramatically lowers CPU usage at the expense of added latency.**

Sound designers and power users may optionally run Kaleidoscope in a stand-alone stereo editor application such as Sony Sound Forge or Steinberg Wavelab in parallel to their multi-track DAW host or as a separate part of the production process. In such hosts, it is recommended to always work with the largest Buffer Size setting. In such scenarios the processing requirements of Kaleidoscope are surprisingly reasonable despite its incredible computational complexity.

Kaleidoscope is designed to offer new possibilities to the world's best sound-designers, composers, and producers who are working on major Hollywood blockbusters and similar projects. A fast computer is assumed in such environments.

In summary, Kaleidoscope can be very demanding on CPU resources when using the maximum settings and small Buffer Sizes. When using these settings, the best user experience will be found by using the newest multi-core processors with high core counts, large caches, and high clock frequencies. Intel Xeon-class workstations are an ideal-case example.

Supported Windows Operating Systems: Win 7 32, Win 7 64, Win 8 32, Win 8 64

Supported Mac OS X Operating Systems: X.7.n (Lion), X.8.n (Mountain Lion), X.9.n (Mavericks), X.10.n (Yosemite)

Supported Hosts: Kaleidoscope is compatible with most known VST, AU, and AAX hosts so long as they conform to the plug-in format standard and work with the hardware and operating systems above. Please refer to “2C-Kaleidoscope Tested Hosts.txt” for a specific list of tested hosts. If your desired host is not on this list, please try the demo to verify compatibility with your system, and feel free to email us for additional information.

Installation & Authorization

Overview

Before installing, first quit all running audio applications.

Installation of the plug-in is subject to accepting the End-User License Agreement that you can read during installation. The installation process is as follows:

1. Download, and unzip the Kaleidoscope installer and the 2CAudio Resource Library
2. Run the Kaleidoscope installer
3. Copy and Paste your Serial Number into Kaleidoscope to authorize it
4. Link Kaleidoscope to the 2CAudio Resource Library
5. Enter your User Name and User Info

Windows Installation (32bit & 64bit)

To install the plug-in simply launch “setup.exe” and follow the instructions. You will have the option to install either or both the 32-bit VST version and/or the 32-bit AAX version. If you are using a 64-bit OS, you will also have the option to install the 64-bit VST and/or the 64-bit AAX version(s). Note that in order to load the 64-bit version of the plug-in you must use a 64-bit audio host. If you have a 64-bit VST host we recommend installing the 64-bit version for use with this host.

Note: In rare cases some users have reported that clicking on the installer application (“setup.exe”) does not successfully launch the installer. The reason for this is generally that the installer has been downloaded to a network drive, a shared local drive, or is saved deep in the folder hierarchy of the users systems. In such cases Windows may block the launch of our installer for “security” reasons. If this occurs, simply temporarily move the installer folder to either your “Desktop” or to your “Program Files” folder and run the installer from this location. This will solve the issue. Upon successful installation you may move or delete the installer from this temporary location as desired.

Mac OSX Installation (32bit & 64bit)

To install the plug-in double-click on “Install 2C-Kaleidoscope.mpkg”. A standard OSX Installer will launch. Follow the installer instructions. By default, VST, Audio Units, and AAX versions are installed. If you would like to install only one particular format, please follow the installer instructions on the readme page. Both 32-bit and 64-bit versions of all formats are installed.

Installing the 2CAudio Resource Library

In addition to the Kaleidoscope plug-in itself, your order also contains a “2CAudio Resource Library” download folder. This folder contains a large library of resources that are used by Kaleidoscope including images, tuning scale files, and waveforms. Resource library expansions such as *Galbanum Architecture Volume One* are also installed to this location. The total size of this folder’s contents may grow to be many tens of gigabytes or larger over time, and therefore we allow

this folder to be located anywhere on your system including network drives. To install this folder, simply unzip it and move it to wherever you would like to keep it.

Authorization

To authorize the product you need to enter the serial number that you were provided when you purchased the product. Upon completion of your order through the Galbanum web store, you should have received an email receipt that confirms your purchase. This email will contain your personal Serial Number. If at a later date you need to re-download your product or re-enter your serial number, you may find both the download link and your serial number in the order details page for the given order. You will find this in your order history in the Galbanum web store.

Once you have located your serial number, select and Copy the entire serial number. Pressing “Control-C” (Windows) or “Apple-C” (Mac OSX) on your keyboard will copy your selection. Alternatively you may find the Copy command in your OS’s Edit Menu or contextual menus.

After copying your serial number, launch your host application and instantiate Kaleidoscope. Upon loading the plug-in in your host application for the first time the plug-in will automatically display the Info Page of the GUI. Click on the Serial Number field, which should currently be displaying the message “Click here to enter your Serial Number...”

Paste your Serial Number into this field. Pressing “Control-V” (Windows) or “Apple-V” (Mac OSX) on your keyboard will paste your previously copied selection. Alternatively you may find the Paste command in your OS’s Edit Menu or contextual menus.

Press Enter/Return on your keyboard. If you have correctly entered a valid Serial Number, the product should become authorized, the status text will change from “Enter Serial” to 'Authorized', and your Serial Number will be hidden and displayed like this:

.....

If the Serial Number is not accepted, the status text will not change, and the text String you have entered will remain visible in the Serial Number field. If this happens please check your Serial Number and try to enter it again. As long as the plug-in remains unauthorized, it will not perform any audio processing. Once it has been successfully authorized, processing will begin and you may navigate to the Main or Browser Page to begin using the plug-in.

Linking Kaleidoscope to the 2CAudio Resource Library

Upon successfully authorizing Kaleidoscope, it must be pointed to the correct location of the shared “2CAudio Resource Library” folder. On the Info Page of the GUI above the authorization area there is a field labeled “Library Location”. Click on this field and point the file dialog to the location of the “2CAudio Resource Library” as you have located it.

The location of the “2CAudio Resource Library” is determined entirely by you. You may move it to a new location at a later date if you choose, but you will need to re-link Kaleidoscope to it by performing this step again.

Entering User Info

Above the Library Location field are found the “User Name & Info” fields. Please enter your name into the left field, and optionally any information you might like to include, such as a company name or website in the right field. If you create your own custom presets, this information will be saved into the presets. This information is also displayed on the Browser Page of the product GUI, and if you share your presets with others this is a good way to gain recognition and credit for your work.

File Management

This information is really only needed when changing operating systems or upgrading to a new computer if you have created custom User Presets.

Windows File Management

All Kaleidoscope-related program files and plug-in resources are saved in a folder named “/Program Files/2C-Audio/Kaleidoscope” that is created by our installer. This folder does not contain any user data.

The 32-bit VST shell, 64-bit VST shell, and AAX shells, are saved in the hosts’ respective plug-in folders. These files serve to connect the hosts to the plug-ins and resources found in “.../Program Files/2C-Audio/Kaleidoscope/”. They do not contain any user data.

Presets are stored in “.../Users/**you**/My Documents/2C-Audio/ Kaleidoscope /Presets/” by default. You may change this install path to any location you like including locations on non-system disks.

The shared “2CAudio Resource Library” folder location is determined entirely by you.

This information is really only needed when changing operating systems or upgrading to a new computer if you have created custom User Presets.

OSX File Management

The Kaleidoscope Audio Unit shell is installed here:

.../Library/Audio/Plug-Ins/Components/

The Kaleidoscope VST shell is installed here:

.../Library/Audio/Plug-Ins/VST/

The Kaleidoscope AAX shell is installed here:

.../Library/Application Support/Avid/Audio/Plug-Ins/

These files serve to connect the hosts to the plug-ins and resources found in “/Library/Application Support/2C-Audio/Kaleidoscope/”. They do not contain any user data.

All supporting files, presets, and plug-in resources are installed here:

.../Library/Application Support/2C-Audio/Kaleidoscope/

User Presets are stored in “.../Application Support/2C-Audio/Kaleidoscope/Presets/User/”.

Uninstalling

2CAudio installers do not create hidden files on your system. Windows users may use Windows’ “Add/Remove Programs” feature or run the uninstaller included within Kaleidoscope’s setup folder. To uninstall on OSX you simply have to delete the plug-in files and the associated resource files. The location of these files is described in the previous section. If you intend to reinstall Kaleidoscope again at any point in the future, and if you have created your own User Presets, be sure to back these up before deleting your installation.

The GUI

The Navigation Bar

The Kaleidoscope GUI is segmented in three main pages: the Main Page, the Browser Page, and the Info Page. At the top of all three of these pages is found a common area called the Navigation Bar. This area has two functions: to allow navigation to the various GUI pages, and to handle global preset and file needs. The navigation bar is shown below.



Page Navigation

On the left side of the Navigation Bar is found the Page Navigation area. To switch the Main Page, click on Main. To switch to the Browser Page, click on Browser. To switch on the Info Page, click on Info. Any questions? No? We didn't think so.

Preset and File Actions Area

On the right side of the Navigation Bar is found the Preset and File Actions Area. This area allows you to change, load and save presets. The controls have the following functions:

- **Previous Preset:** changes the preset to the previous available preset—sorted in alphabetical order—in the active preset folder or the previous available folder when at the beginning of the list.
- **Next Preset:** changes the preset to the next available preset—sorted in alphabetical order—in the active preset folder or the next available folder when at the end of the list.
- **Active Preset:** displays the name of the currently active preset. More accurately, it displays the name of the last loaded preset from existing factory or user presets. If you make changes to the preset settings, the preset name will not change automatically, and there will be some differences between the settings stored in the Active Preset name and the actual current settings. Note that host applications save all current settings exactly as they are, so your edits will be preserved. However if you return to a project at a later date you may see the Active Preset name displayed as one of the factory presets, and you may try to load this preset into another project to achieve the same sound without realizing that you have made substantial changes to the preset at an earlier date. Therefore, if you edit the preset substantially and you think you may like to use these settings at a later date in another project, it is a good idea to save the settings as your own custom User Preset. *Clicking anywhere in this area accesses the custom preset panel and allows a single preset to be selected from anywhere within the hierarchical preset directory.*
- **Active State:** 2CAudio plug-ins store two independent sets of parameter settings: A and B. Clicking on the Active State button switches between them and allows them to be compared. This feature has several uses:

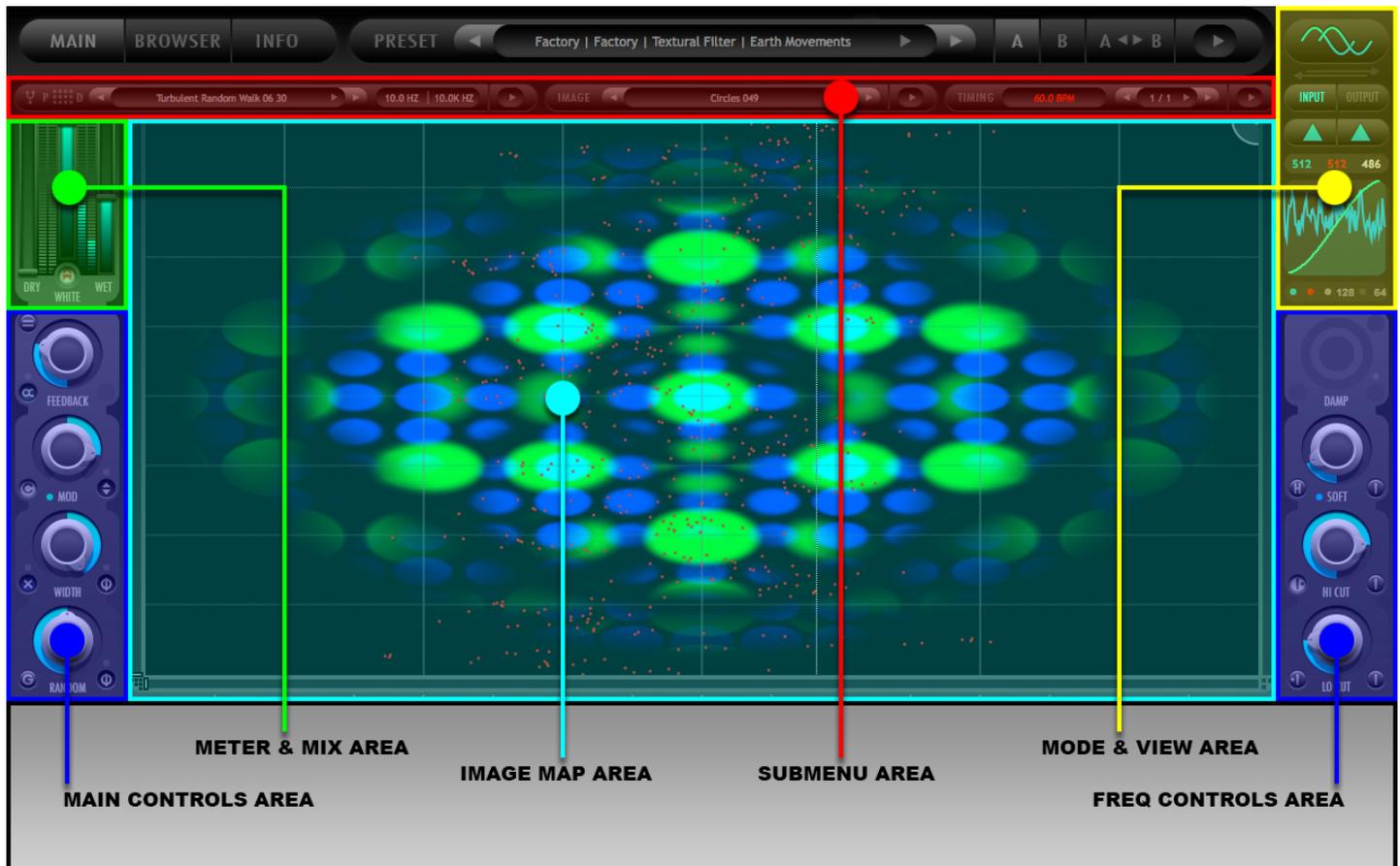
- Saving potential preset matches while browsing presets: if for example you are trying different presets in the Browser Page and you find a good candidate but you would like to try some others, you can quickly copy the candidate to the inactive State and continue browsing. When you find another one that you like, you can compare it to the previous candidate to decide which is better.
- Comparing changes made to presets with the original preset: you may use the same technique to check changes you are making to a preset by first copying the preset to the inactive State and then performing your edits and finally switching States again to compare the results.
- Blind listening tests: oftentimes engineers succumb to the weakness of using their eyes or minds more than their ears, and particular numerical values or aesthetic GUI settings are chosen over simple listening preference. Blind listening tests can cure this problem. This can be accomplished by saving two different presets in State A and State B, closing your eyes, and clicking an unknown number of times on the Active State button such that you no longer know if you are using State A or State B. At this point keep your eyes closed and listen critically while you slowly cycle back and forth between settings while comparing the results with your ears only. Once you have chosen the State that sounds the best, open your eyes and look at your selection. Repeat this process for confirmation if desired. This is a powerful technique.
- **Copy State:** Copies the settings of the Active State to the inactive state. For example if State A is active, the settings are copied to State B. This allows easy comparisons between states to be made as described above.
- **Preset Actions Panel Button:** Accesses the Preset Actions Panel and gives access to the following preset file operations:
 - **Replace Preset (Overwrite):** saves the current plug-in state to last loaded Preset File location and overwrites the existing file. This is intended to offer a quick way to make edits to presets during the preset design stage.
 - **Save Preset to My Presets:** saves the current plug-in state to the “My Presets” folder in the User folder and adds it to the preset list.
 - **Save Preset to Folder:** saves the current plug-in state to a different folder location other than the “My Presets” location. By default, the file dialog points to the folder of the currently active host project file if this information is available. This is useful for archiving favorite presets together with host projects and for sharing a given preset with peers on either local or global networks.
 - **Load Preset:** loads a single preset file into the current plug-in state. This is useful for loading presets that were saved and/or shared with the “Save Preset to Folder” action.
 - **Import Preset to My Presets:** loads a single preset file into the current plug-in state and copies this preset into the My Presets SubFolder found in User. This is useful for importing single presets that were shared by users on local networks or web forums.
 - **Import Preset SubFolder to User:** copies a single SubFolder of preset files from into the User Package and rebuilds the preset browser and menu lists. This is useful for importing a single folder of presets that were shared by users on local networks or web forums.
 - **Import Preset Package:** copies an entire directory of SubFolders containing preset files into the root of the preset directory and rebuilds the preset browser and menu lists. This is useful for importing commercial Preset Expansions that were purchased from our web store or developed by third parties.

The Main Page

Layout

Kaleidoscope's GUI is divided into three pages: the Main page, the Browser page, and the Info page. The Main page design is deceptively simple at first glance, but offers extreme control over incredible complexity. The Main Page is conceptually separated into five sections: the Submenu Area, the Image Map Area, the Meter & Mix Area, the Mode & View Area, and two Controls Areas.

- **The Submenu Area** contains parameter displays and controls for Tuning, Image, and Timing. These controls are of critical importance to the sound of a given preset and each sub menu area provides an Action Panel which exposes additional parameters and settings as well a Resource Library Browsers.
- **The Image Map Area** displays the currently selected Image Map.
- **The Meter & Mix Area** display control gain levels within Kaleidoscope.
- **The Mode & View Area** displays buttons to control Kaleidoscope's various modes and view options.
- **The Two Controls Areas** offer parameters to further adjust and fine-tune the sound of a given preset.



Note: The 2CAudio products are “skin-able”. 2CAudio may offer multiple skins for the same product. Third parties may also develop additional skins. Thus the interface may appear differently than as described in this manual. This document describes the “Moonlight” skin and uses it with the Cyan color variation for screenshots.

The Info Page

The Info Page contains three main functional areas:

- **The Tuning Info Area** displays information and notes about the tuning used for the currently active preset.
- **The Preferences Area** contains menus and controls to adjust various preferences and options related to Kaleidoscope's behavior and appearance.
- **The Karma Boost Area** contains the product authorization functionality and displays authorization status and version number.

The screenshot shows the Kaleidoscope software interface. At the top, there are navigation tabs: MAIN, BROWSER, INFO, and PRESET. The PRESET tab is active, showing a list of presets: Factory, Textural Filter, and Strange Attactor. Below the navigation is a red bar labeled 'TUNING INFO' containing a waveform and text: 'N Exponential divisions of ratio range R from 1.0 to R. Exponential Base curve B. (Author: Andrew Souter, 2011)'. Below this is a cyan bar labeled 'PREFERENCES' with sections for GUI OPTIONS, PREFERENCES, AUDIO OPTIONS, QUALITY & EFFICIENCY, REALTIME, and OFFLINE. Below the preferences is a green bar labeled 'KARMA BOOST' showing 'Version 1.0.0 [A332] x64 AVX | Authorized' and a progress bar. At the bottom, three labels with lines pointing to the corresponding areas: 'PREFERENCES AREA' (cyan), 'TUNING INFO AREA' (red), and 'KARMA BOOST AREA' (green).

Mouse and Keyboard Interface Gestures

Mouse-Click Behaviors

- **Click-Drag:** selects a knob and changes its value at normal rate
- **Shift-Click-Drag (Knobs):** selects a knob and changes its value at a slow, fine-tune rate
- **Shift-Click-Drag (Image Map, Sliders & Circle):** snaps offset values to the X & Y grid values
- **Control-Click (Win):** sets the knob value to the default value for the parameter
- **Command-Click (OSX):** sets the knob value to the default value for the parameter
- **Double-Click:** opens the text-entry edit box for the selected knob or value box

Mouse-Wheel Behaviors

- It is possible to use the mouse wheel to change a knob value. To do so, mouse-over the knob you want to control, and then simply use mouse wheel.
- Holding down the Shift key on the keyboard while using the mouse wheel, will change the values at a **very** slow, fine-tune rate

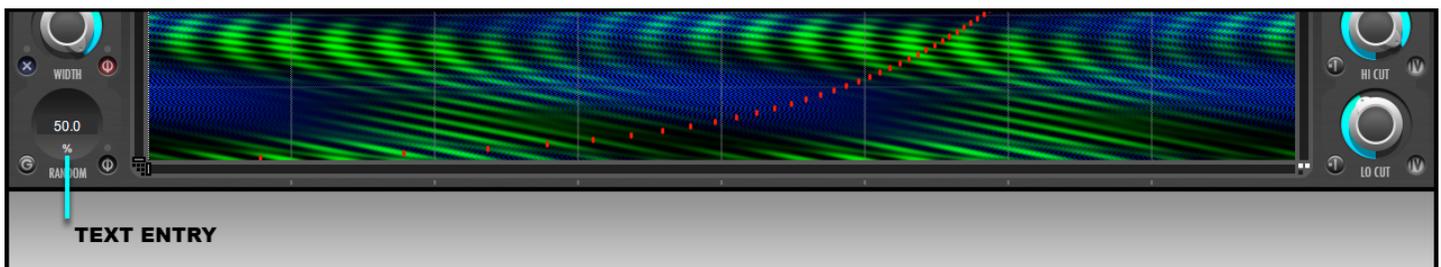
Keyboard Shortcuts

In some hosts you can use keyboard keys to change knob values and switch presets in the Preset Browser. To activate keyboard control, first click somewhere in the Preset Browser window or on the knob you wish to control. The keyboard shortcuts as follows:

- **Preset Browser Keyboard Shortcuts**
 - **Left / Right / Up / Down Arrow Keys:** moves preset selection one step in the respective direction
 - **Page-Up / Page-Down:** shifts the column display one step left or right
 - **Home:** Selects the first preset
 - **End:** Selects the last preset
- **Knob Keyboard Shortcuts**
 - **Up / Down Arrow Keys:** changes value by 2%
 - **Left / Right Arrow Keys:** changes value by 0.1% (fine tune)
 - **Page-Up / Page-Down:** changes value by 10%

Text Entry

Double-clicking on a control (knobs, sliders, and Mix control) opens the Text Entry mode for the control. Floating point values may be entered with any desired decimal precision. The initial precision displayed in the text entry mode when first opening it is formatted to fit within the given text entry field. Full precision is stored in preset files and used in the DSP calculations. Parameter units (i.e. Seconds, Hertz, Decibels, etc.) are displayed below the text entry field.



Getting Started

What Is Kaleidoscope?

It's a trip. Really! Kaleidoscope is an entirely new class of effects processors and is one of the most unique signal processing effects to come to market in recent history. At the risk of sounding immodest, it may very well become one of the defining sounds of 21st century. Kaleidoscope is a massively parallel bank of physically modeled resonators that can be tuned completely arbitrarily with scientific precision and are dynamically modulated over time via over two million points of automation in the form of two independent image maps. In simplistic terms, Kaleidoscope uses pictures to control sound!

So what does that mean? To understand the answer let's discuss these things separately:

- What is a resonator?
- What is an image map and how is it used to modulate the sound?
- Who cares about custom tunings and what precision they are?

What Is A Resonator?

A resonator is a specialized filter. A filter is physical device or a process that changes the spectrum (frequency component amplitude) and/or phase (timing information) of an input signal. A filter works by attenuating different frequencies by unequal amounts. A Low-Pass filter retains frequencies below the cut-off point while attenuating frequencies above the cut-off point. It is also known as a Hi-Cut filter for this reason. This is a very simple example.

A resonator is a device or system that naturally oscillates at some frequencies with greater amplitude than at others. Resonators are used to either generate specific frequencies or to select specific frequencies from a signal. "Select" in this context means that the resonator will attenuate all other frequencies except the so-called "resonant frequencies". Thus a resonator is a type specialized filter; it passes the resonant frequencies and cuts all other frequencies. Therefore if a resonator is excited by some form of broad-band, spectrally-dense input signal such as a burst of white noise, or an impulse click, or the transients of a physical strike or pluck, or the noisy breath of a woodwind player, or the consonants of a vocalist, the resonator will quickly attenuate all non-resonant frequencies while allowing the resonant frequencies to sustain, or ring, or oscillate significantly longer than the non-resonant frequencies.

Consider the analogy of the prism. A prism is a triangular piece of glass that can be used to separate white light into its constituent spectral colors (i.e. the colors of the rainbow). The white light that enters the prism is equivalent to a broadband audio signal. "White" in the context of white light, as well as white noise, refers to the fact that the light or the noise signal has equal amplitude at all frequencies within the spectrum. The prism breaks apart the white light in a way that clearly demonstrates the presence of the various spectral components (i.e. colors). If the prism were able to selectively attenuate or cut some of these colors while at the same time passing or augmenting others, then the prism would be functioning like a resonator in many ways. This would be an even more accurate metaphor if the prism continued to output -- i.e. ring or oscillate at -- the selected colors for some time after the white light input was stopped. In other words, if we could build a prism that passed only red and yellow while cutting all other colors and it made red and yellow ring or sustain for some period of time after we turned off the input, this would be very similar to the behavior of a resonator.

In the real physical world, all musical instruments use acoustic resonators that filter sound waves at specific frequencies. Resonators are a fundamental building block of the mathematics and physics of physical sound generation and filtering. **Every musical instrument has resonators!** Some generate the sound directly, such as the Strings in Stringed instruments, the head of a drum, the wooden bars in a xylophone, and the pipes in an organ. Some modify the sound by enhancing particular frequencies, such as a piano soundboard or the sound box of a guitar or violin. Organ pipes, the bodies of woodwinds, and the sound boxes of Stringed instruments are also examples of resonators. There is in fact an

entire branch of musical instrument synthesis research and development called **Physical Modeling** that deals specifically with these topics and related ideas.

Perhaps somewhat counter intuitively at first, even real-world acoustic spaces such as anything from a small closet, telephone booth, or automobile, to large spaces such as concert halls and cathedrals can also be considered a type of resonator called an **Acoustic Cavity Resonator**. Reverberation is in fact a form of resonance, albeit a very, very complex one with the number of resonant frequencies theoretically in the billions for real large spaces such as concert halls. Yes indeed, 2CAudio's Aether, B2 and Breeze plug-ins may be thought of as extremely complex resonators! In reverberation however, we go to great lengths to be sure all colors (frequency components) are resonated equally and the resulting output light (spectrum) remains white to borrow the prism metaphor again.

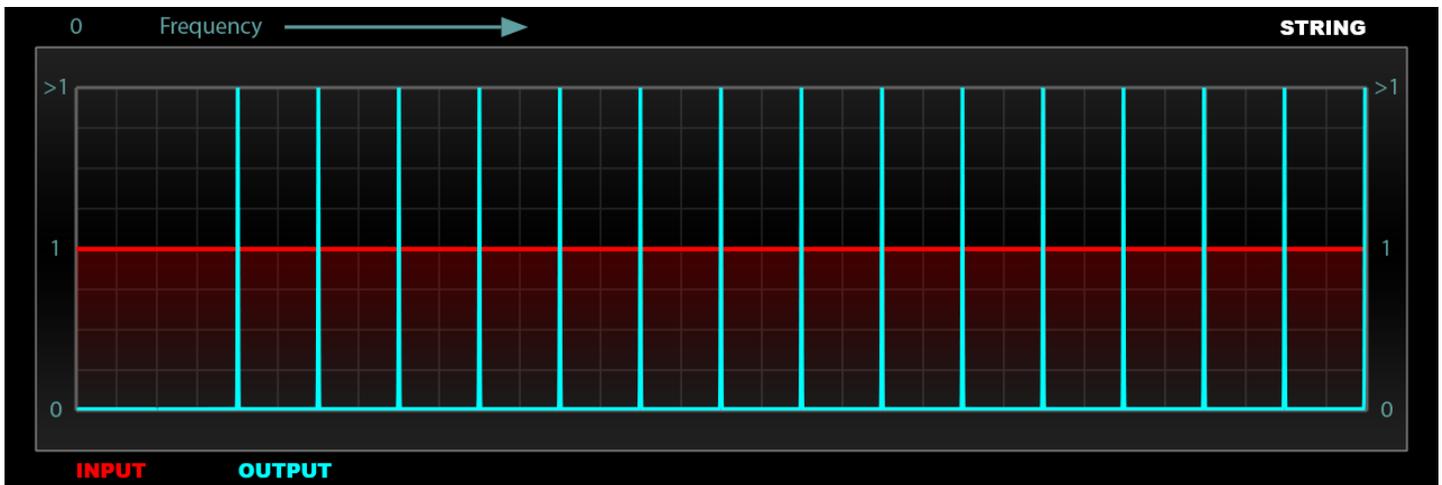
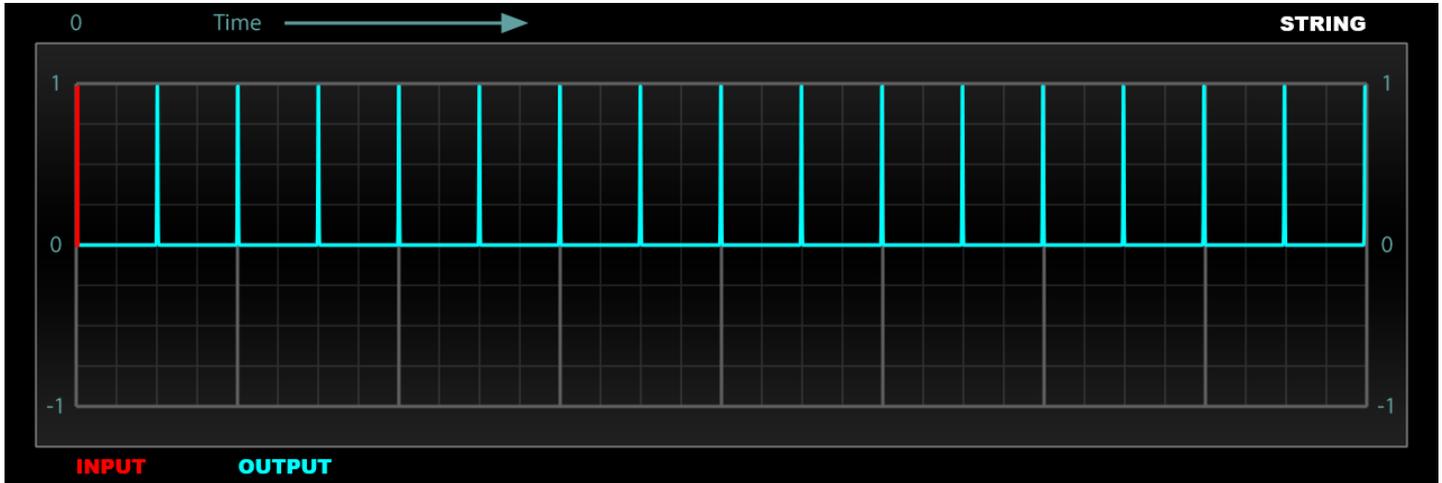
Kaleidoscope in contrast, is all about color! Here we want to intentionally color the sound in very specific and purposeful ways as you will see and hear. Kaleidoscope functions much like a selective time-varying prism by reacting to the musical input in very specific ways: drastically augmenting certain frequencies and drastically attenuating others. Kaleidoscope offers two primary types of resonators: Strings and Springs.

Strings

The Strings found on all Stringed musical instruments are simple resonators. Stringed instruments work by applying some excitation force in the form of a hammer strike, or a plectrum pluck, or the rubbing action of a bow to the String. This sets the String into motion and it begins to oscillate. In the spectral sense, the excitation action supplies a form of broadband noise. This can be thought of as something like an impulse, a short transient, or some form of enveloped noise in the case of bowing. The interesting thing to note is that this input source is not pitched; it is in fact similar to white noise in that it contains all frequencies equally. More accurately, perhaps there is some gentle low pass filtering action or spectral tilt to make it approach something more like pink noise with less energy in the extreme high frequencies, but the fact remains that there is generally no distinct pitch or spectral peak in the excitation input. This is intuitively obvious by considering the fact that the excitation action is generally the same for all Strings on Stringed instruments. The same hammer action strikes C#6 as well as Bb1 in the case of the piano for example. Therefore it is not the excitation force that creates the pitched oscillation; it is the physical properties of the String itself, which acts as a resonator.

A String oscillates at a particular frequency that is determined primarily by its length, mass, and tension. From the physics perspective the String acts as a resonator: its three physical variables will define what fundamental frequency it will resonate at. It converts the broadband flat-spectrum input energy into a signal with a very distinct pitch, or resonant frequency.

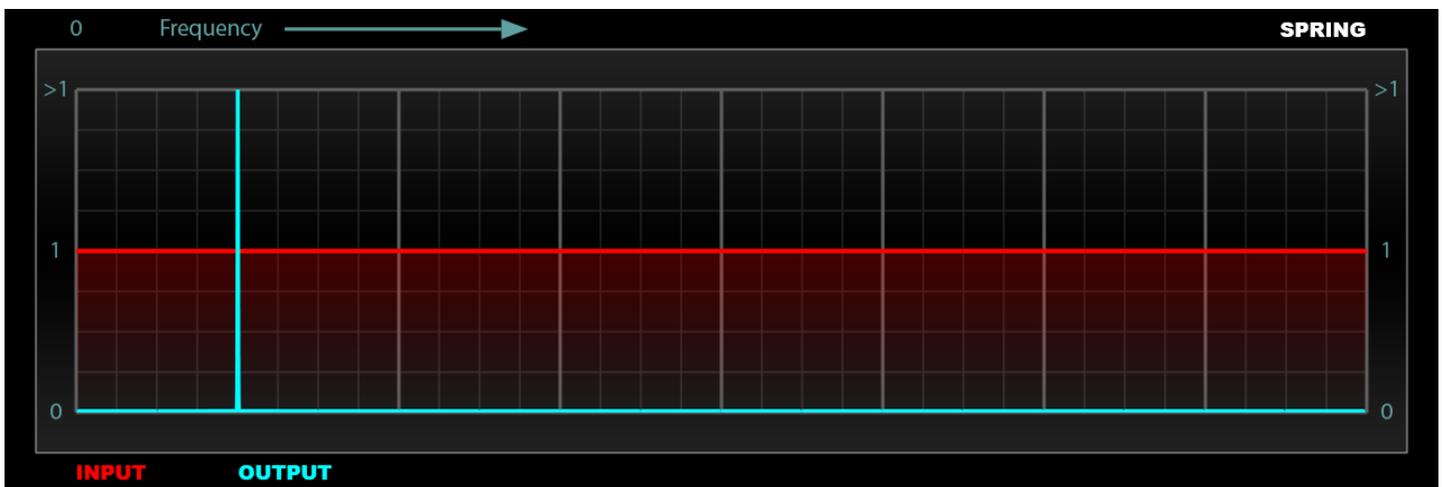
*Many types of resonators in the real world actually resonate at more than one frequency, and will have a set of multiple resonant frequencies that are determined by the object's physical attributes and geometry. This is true of Strings, and in this case it turns out that these resonant frequencies follow a very simple rule: **Strings follow the harmonic series** (more or less). Thus if the String oscillates at a fundamental frequency of 100Hz, it will also simultaneously oscillate at integer multiples of this frequency (i.e. 200, 300, 400, 500 etc.). Shown below are the impulse response and spectrum of a String model. Note the impulse response is simply a series of delayed copies of the original impulse and the spectrum is that of a narrow comb filter which produces peaks at the fundamental frequency as well as all integer multiples: i.e. harmonics.*



Summary: String resonators contain harmonics. A single String resonates all integer multiples of the fundamental frequency. Strings are much more complex spectrally than Springs.

Springs

Spring resonators are the simplest variety of resonators. Springs oscillate at only one single frequency: the fundamental frequency. Spring resonators do not contain harmonics. A **Mass On A Spring** model and a **Pendulum** are two examples of such systems. This type of system is called a **Simple Harmonic Oscillator** in physics, and its behavior is called **Simple Harmonic Motion**. Shown below are the impulse response and spectrum of a Spring model without damping. It is identical to a sine wave.

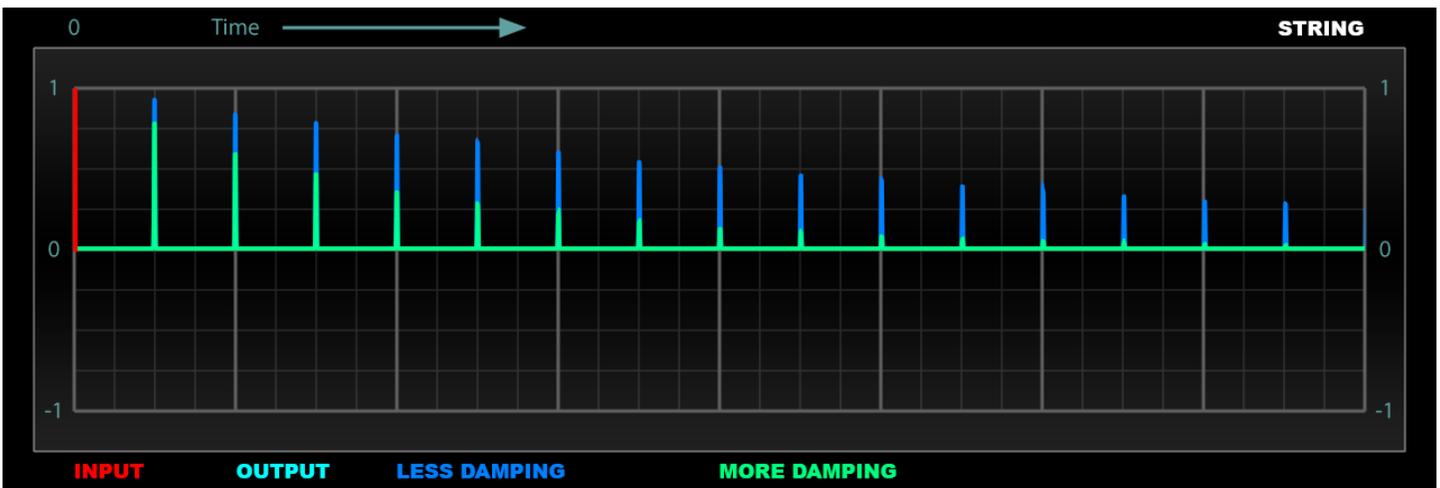
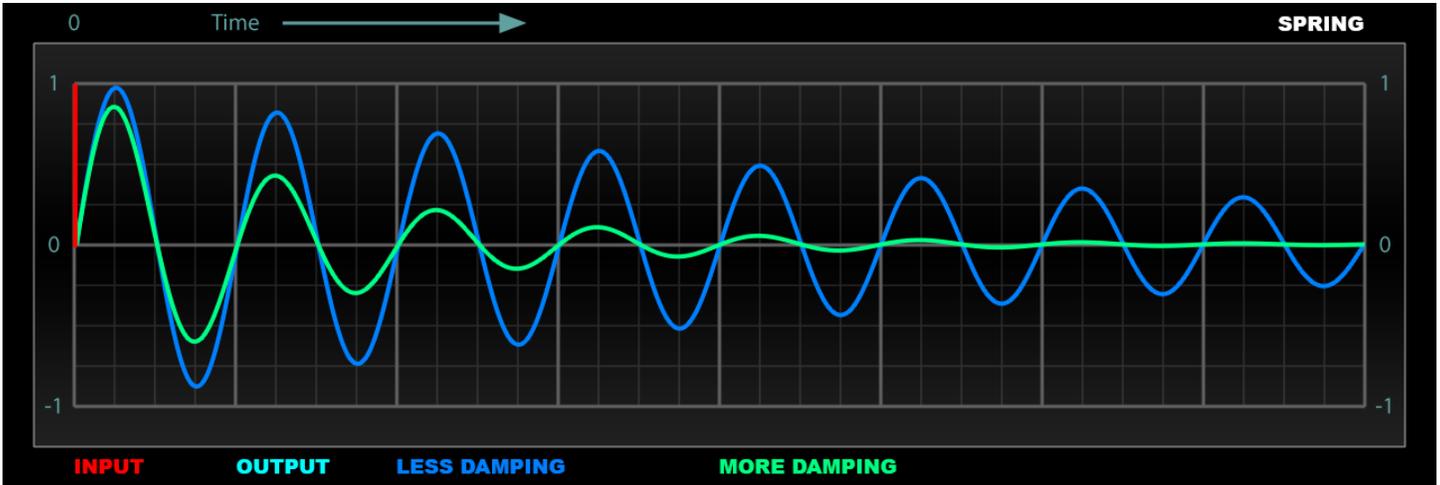


Due to their single resonant frequency, sinusoidal nature, Springs are ideally suited to create any arbitrary spectrum by summing a large number of them together and precisely tuning each individually to any desirable tuning. **Fourier** and **Modal Analysis** methods can effectively be used to accurately model the resonant behavior of much more complex objects and match their spectrums. This can be thought of as a form of additive synthesis. An additive synthesizer with enough resolution in both time and frequency can effectively perfectly synthesize any sound.

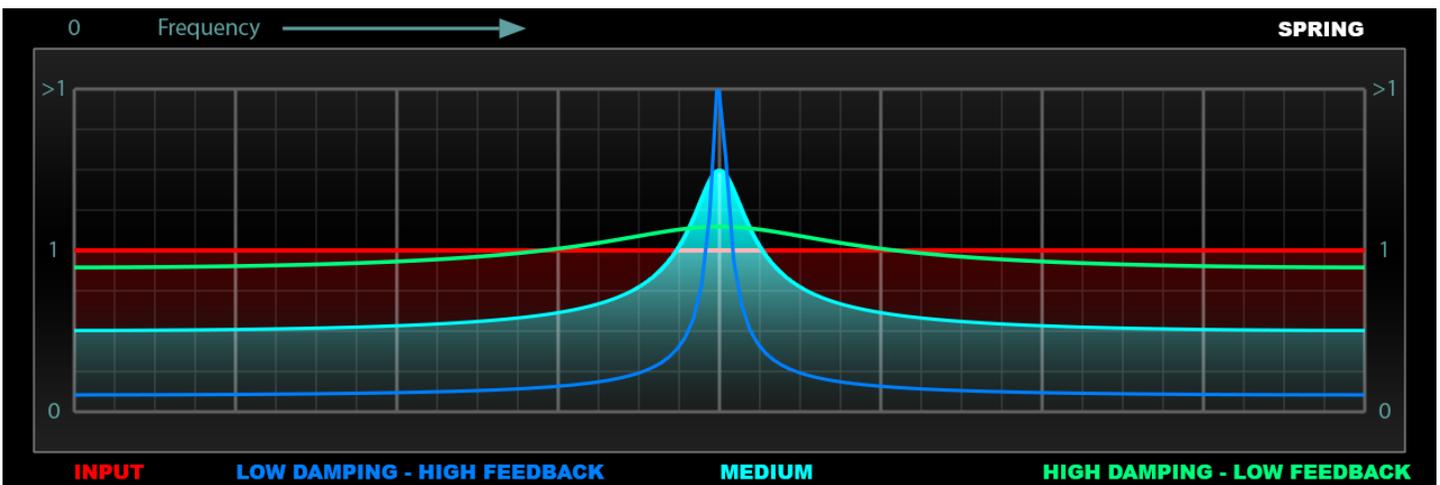
Summary: Spring resonators do NOT contain harmonics. A single Spring resonates only at fundamental frequency. At high enough feedback, Springs effectively produce sine waves. At low feedback they produce narrow band filter effects. Multiple Springs can be used as a form of additive synthesis.

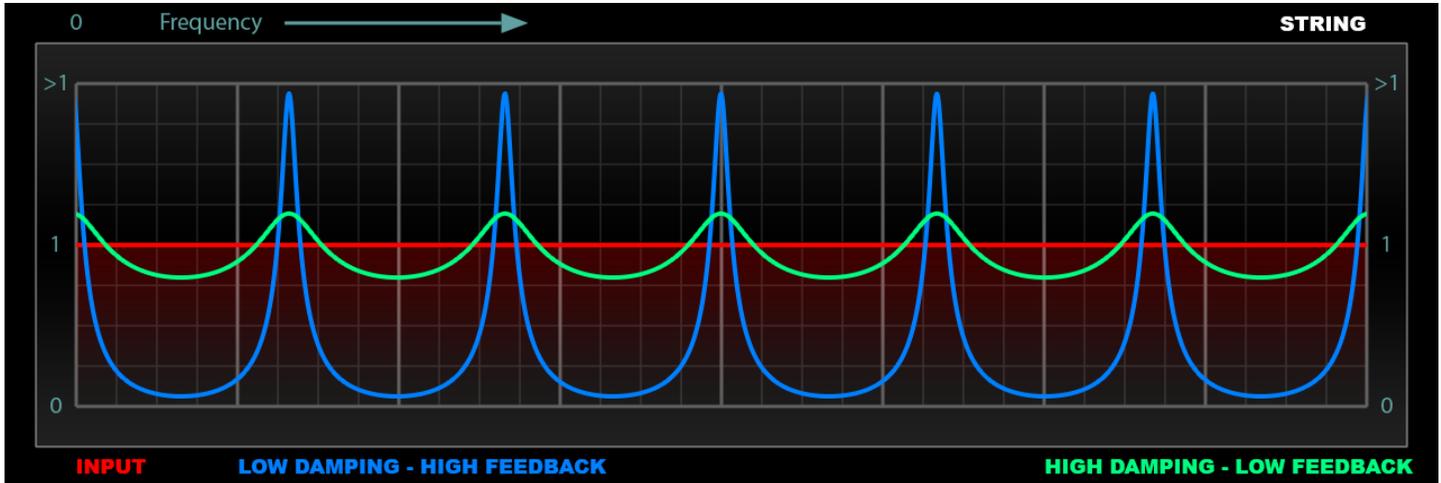
Damping & Feedback

Resonators in the real world do not continue to oscillate forever; they lose energy over time. This is called **Damping** in physics and is caused by things such as friction. This type of action creates exponential decay in the amplitude of the oscillation of the resonator. Shown below are examples of the behavior for damped Strings and Springs. Springs produce exponentially decaying sine waves. Strings produce an exponentially decaying series of delays.



Damping has an effect on the resulting spectrum of the resonator as well. Springs pass perfect sine waves and nothing else only when the Spring continues to oscillate forever. The same is true of Strings as well: Strings pass a perfect harmonic series and nothing else, only when there is no damping. This changes when damping is introduced. Spectrally as the damping increases, (i.e. feedback decreases), the perfectly narrow single-frequency peak gets wider and starts to behave like a narrow band-pass filter or a resonant low pass filter. The same behavior occurs in Strings as well: when damping is high the width of spectral peaks increases and the attenuation of the non-resonant frequencies decreases.

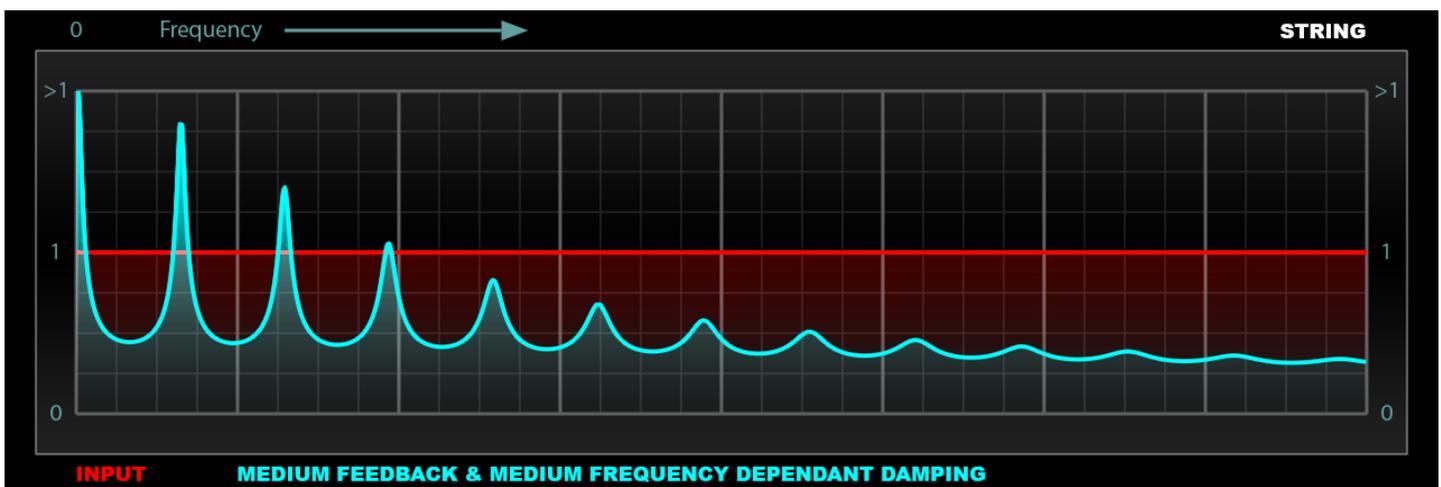




The terms **Damping** and **Feedback** are related. They are two ways of naming the same phenomenon or behavior, but they are exactly inversely proportional; if Damping is high, feedback is low and vice versa. Physics literature typically speaks in terms of damping as we have above so far; however, in Kaleidoscope we actually use the term Feedback for several reasons. The most obvious reason is that all resonator types other than Springs, resonate at multiple frequencies as discussed. Strings for example, have harmonics. These individual harmonics do not necessarily have the same amount of damping and feedback. In fact, in almost all real physical systems, including a plucked String as well as a concert hall, high harmonics generally have much lower feedback (i.e. higher damping) than the fundamental frequency. There is a **Frequency Dependent Damping** effect that generally happens in real-world systems. The audio industry generally refers to this phenomenon as “Damping”, like for example the “Damp” controls in our own Aether, B2, and Breeze products. Therefore when we are discussing these topics in Kaleidoscope we use the following definitions:

- **Feedback:** the base level of feedback or resonance at the fundamental frequency (similar to RT60 in reverb)
- **Damping:** the relative decrease in feedback (or decay time) for some specific frequencies/harmonics/partial

In digital audio String models can be implemented as a simple **Feedback Comb Filter**. Frequency-dependent damping is achieved by putting a filter of some variety (generally a low pass filter) into the feedback path. With the proper parameter settings this simple model can create very realistic sounding String sounds. The first exploration of this general idea was the Karplus–Strong algorithm in 1983. In Kaleidoscope, when using the String resonator modes, a liberal amount of frequency dependent damping is almost always desired. Shown below is the resulting spectrum of such a model:



Damping, of the relative frequency dependent variety, has no meaning if there is only one resonant frequency as in the case of Springs. Therefore in Kaleidoscope, the Damp control is disabled when using the Spring modes.

Summary: High feedback produces more resonance and longer decay times. Spectral peaks at the resonant frequencies become taller and narrower, and stop band areas are more and more attenuated as feedback increases. Low feedback settings act more like band-pass filters and are not strongly resonant. Frequency dependent Damping achieves variable feedback amounts at different frequencies, and generally this is used to make high frequencies decay faster than low frequencies, as is commonplace in nature.

Driving

Resonators are specialized filters; they produce no sound of their own. Therefore to achieve an audible output, a resonator must be given some form of input energy. As described above, in the physical world, this input typically comes in the form of some form of impulsive energy input such as hammer strike of a plectrum pluck or some form of bowing action. In physical modeling synthesis the input is generally an impulse or a short burst of noise. These inputs excite the resonator and cause it to magically transform the transient broad-spectrum input into a musical output.

In Kaleidoscope we take this concept much further and allow the user to feed any variety of audio signal into Kaleidoscope's massively parallel resonator bank. The incoming audio signal becomes the driving force supplied to the resonators. The output of this system is an incredibly complex, yet strangely familiar, combination of the characteristics of the input signal, the resonator settings, and the action of the Image Maps, which we will discuss in the next section.

Feeding Kaleidoscope percussion loops for example is quite similar to the excitation forces that rhythmic hammer strikes supply to the resonators in instruments such as piano, xylophone, lute and other similar instruments. Feeding Kaleidoscope some variety of slowly enveloped noise would simulate the action found in bowed String instruments. Feeding it material that is already strongly tonal with long periods of static sustained notes will produce less predictable results where the result will depend heavily upon the spectral overlap of the input signal, the current tuning used in Kaleidoscope, and its feedback setting. In such cases, low feedback settings can be used in Kaleidoscope to achieve incredible dynamic filter effects. Feeding Kaleidoscope signals that are a mixture of transients and more sustained tones, such as a vocal signal, which is a mix of sustained vowels and transient consonants can achieve otherworldly results.

The real magic of Kaleidoscope, however, is found in the fact that we are not dealing with a single String or Spring resonator. We are dealing with up to 512 of them, which can each be independently tuned, independently modulated on input and output, and independently spatialized. This is similar to using your voice or drum loop to conduct a symphony played on a 512-String guitar, where each of the Strings is also independently moving around in space according to your specific choreography!

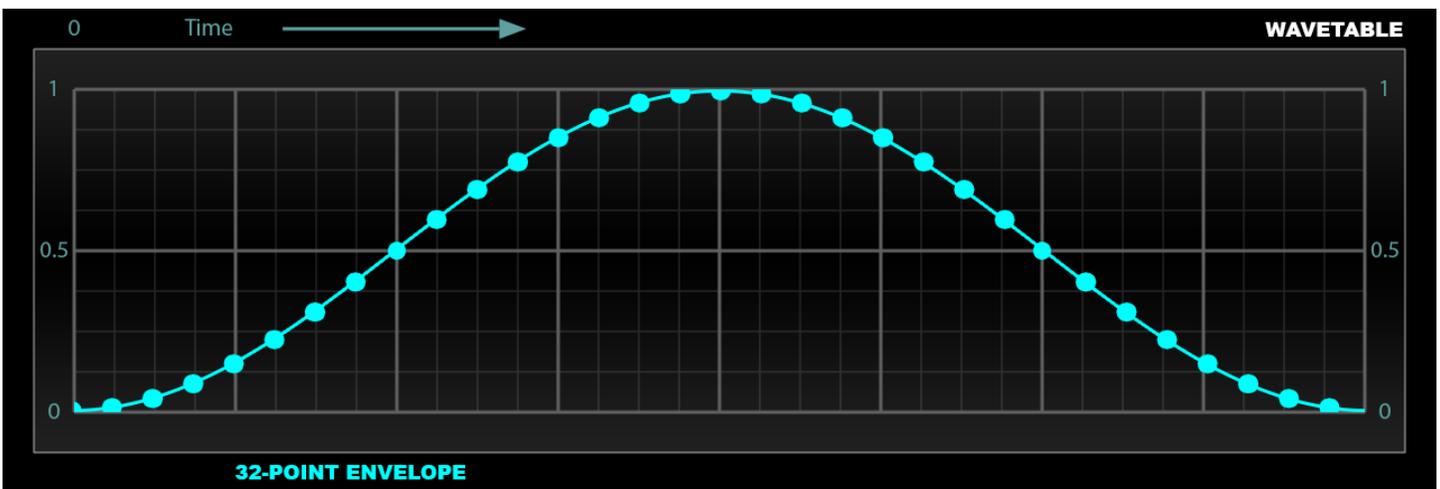
What Is Image Map Modulation?

An Image Map is simply a massive two-dimensional array of modulation/automation data, in the form of a picture, which can be thought of as a collection of many complex envelopes. The image is scanned left to right over time, and each horizontal pixel line represents one envelope.

An envelope is simply a control signal that changes the value of some parameter over time. A simple example is the Attack Decay Sustain Release (ADSR) envelopes found on old analog synthesizers. These can be thought of a simple example of a 4-point envelope with a fifth point assumed to start at zero when time is zero. With such envelopes the user effectively has control over the X and Y position of four points. The algorithm then simply uses linear interpolation to create a straight line between them and this becomes the envelope which is used as a control signal to modulate things like oscillator amplitude or filter-cut-off in an analog synth.



There is no reason to limit an envelope to only 4 points. Many modern music software tools allow the creation of more sophisticated envelopes with many control points. After 10 or 20 points or so however, it becomes quite tedious for the user to manually edit control points to create interesting complex envelope shapes. For complex shapes it can be more beneficial to save the envelope as a wavetable, which is a simply nothing more than a series of evenly spaced X intervals and a Y value for each point. Additionally more sophisticated interpolation can be used to keep the envelope as smooth as possible. A properly interpolated 32-point envelope can already be quite accurate for example.



There is no need to stop there though, wavetables can be much longer to allow for accurate representation of much more complex shapes. Wavetables can easily be 1024-points long for example, and this is exactly what Kaleidoscope offers. (Actually it represents two envelopes superimposed on top of each other: one for the left channel and one for the right channel, as will be explained shortly.)

A single horizontal pixel row in Kaleidoscope represents two 1024-point envelopes, or 2048 points of automation data!

Standard envelopes are usually displayed in a 2D graph like shown above where time is on the X-axis and the Y-axis represents the parameter value, such as amplitude, at that point in time. Kaleidoscope uses pixel brightness as a representation of the Y-value of the envelope. Black pixels represent zero height, zero brightness, and therefore minimum parameter value. Maximum brightness pixels represent, maximum height and therefore maximum parameter value. Pixel brightness in Kaleidoscope can therefore be considered like an elevation map that shows how high each envelope point is. This would be like standing on a plane and looking at a mountain range, where the taller the mountain is at a particular point, the brighter it would be.



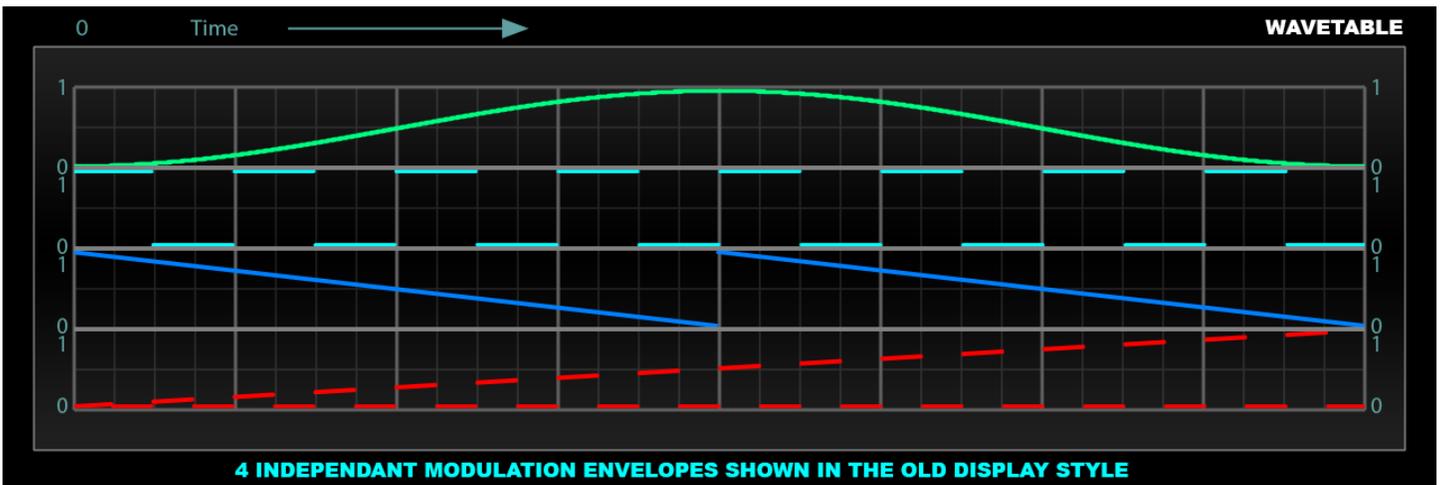
In the above representation the same data is communicated in two different ways: both the height of the top edge of the filled area, as well as the brightness at that particular pixel communicate the same thing. We can get rid of one of these and communicate the same data using only one cue: pixel brightness. Basically we can rotate our modulation curve 90 degrees in the Z axis so that it is coming out of the page and directly at you in 3D space. In our mountain elevation-map analogy, the observer is no longer standing on a plane looking at the side profile of the mountain range. He is now directly above the mountain looking down and can no longer directly see its height profile; he can only see its brightness. Similar to a topographic map, it would look something like the following:



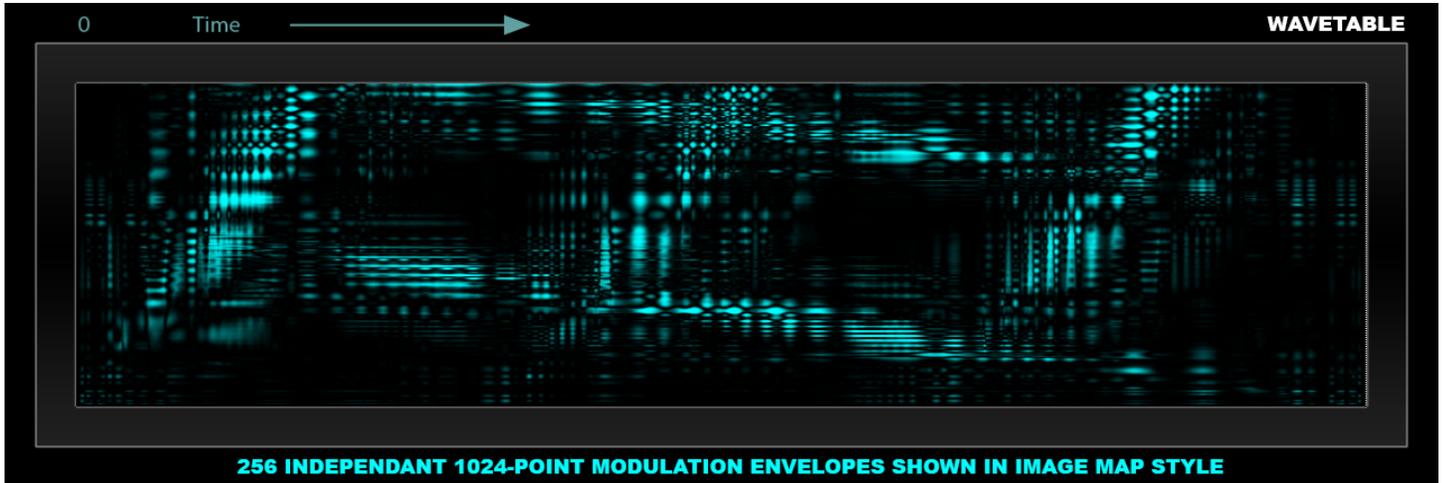
So far we have been talking about a single control envelope. The beauty of the brightness gradient representation is that there is no need to use lots of vertical space to display them. Above, we have used the entire height of this display area to represent one envelope, but this is not necessary. We could communicate the same information using a single pixel row.



Once we start using multiple envelopes it is easy to appreciate how things can quickly get incredibly complicated and messy using old-fashion 2D X-Y view of envelopes. Below is a comparison example of just four independent envelopes:



As you can see, the brightness gradient style used in Kaleidoscope's Image Maps, is much more readable when dealing with multiple envelopes - even a small number of them. Kaleidoscope actively uses up to 512 of them simultaneously, and in such cases, it would be ridiculously impractical if not impossible to display this data in the old style. Thus Image Maps are a highly efficient way to display and manipulate complex modulation data. Just try to represent something like the following using any other method; we dare you!

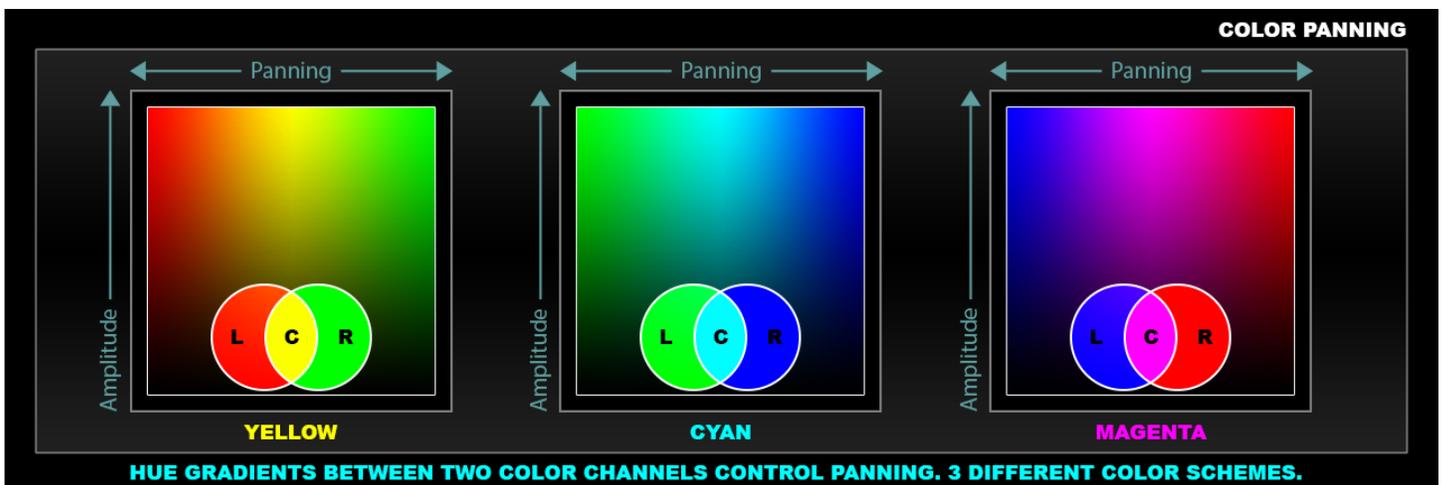


The image above represents roughly a quarter-million points of automation data! This is roughly 1/8th of what Kaleidoscope is capable of controlling!

Kaleidoscope can supply over two million points of automation!

Color

So far we have talked about using brightness gradients to represent amplitude in modulation envelopes. It is also possible to use color (hue) in meaningful ways. Kaleidoscope uses color to effectively control stereo panning. Computer graphics use the three primary colors (Red, Green, and Blue) mixed in variable amounts to create any possible hue. Computer graphics thus use three color-channels. Stereo music and audio productions use only two channels. Therefore we can choose one color channel to represent the left channel, and another to represent the right. The third color channel is not used. This results in three possible color schemes: Yellow, Cyan, and Magenta. The default color scheme for Kaleidoscope is Cyan, and Cyan is used for screenshots in this manual.

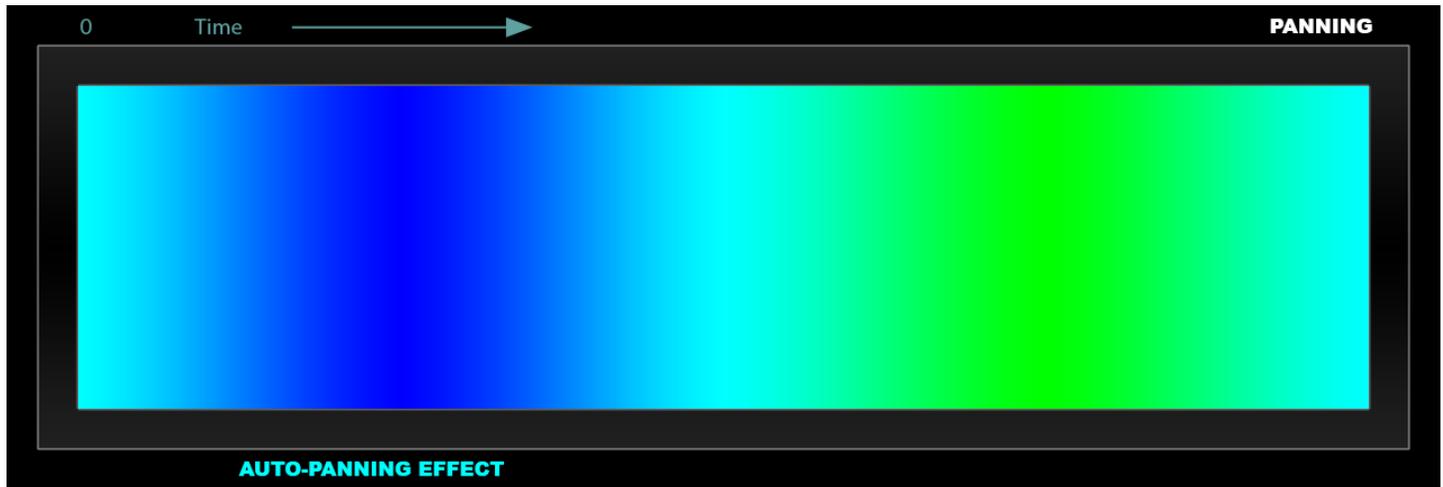


To repeat: one color channel controls the amplitude of the left audio channel, and another color channel controls the amplitude of the right channel. Volume is thus controlled independently for the left and right audio channels and this is exactly how standard gain panning is done in audio mixers. Visually this scheme produces nice hue gradients since computer graphics are based on the additive color principals of light. That is to say, mixing two primary colors such as red and green produces the secondary color yellow. Mixing green and blue, creates cyan. Mixing blue and red creates magenta. Uneven mixes will result in hue somewhere between the two primary colors. Thus panning is independent of

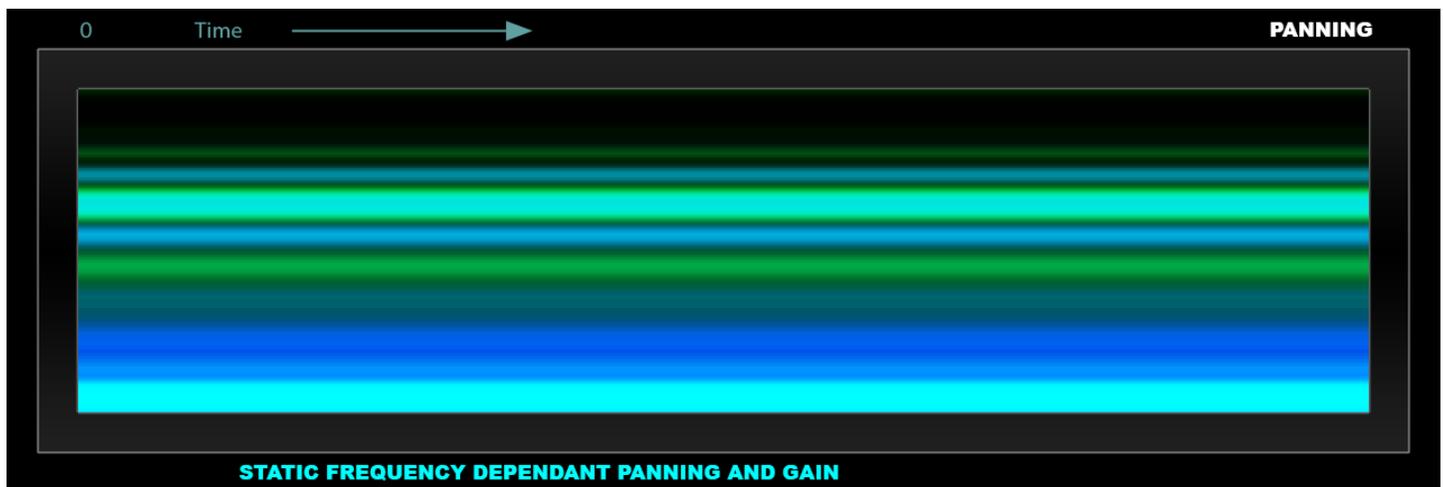
amplitude and as shown above using two color channels can perfectly control both panning and gain and can produce smoothly changing gradients between them.

Kaleidoscope uses brightness to control amplitude and color hue to control stereo panning.

An image map such as the following would produce a dynamic auto-panning effect that would change the panning of all lines by the same amount over time.



An Image Map like the next would produce a frequency-dependent panning and gain effect that is static over time.



From the previous two examples we can generalize some and say that horizontal “banding” in image maps indicates changes over time or space, while vertical “banding” indicates changes over frequency. The important thing to realize here is that gain as well as panning can be both dynamic over time, as well as independent over frequency for each resonator line.

Individual “notes” or voices can move around independently in spatial position and as well as change in gain. This is not possible with traditional MIDI-based synthesizers and effects.

For example:

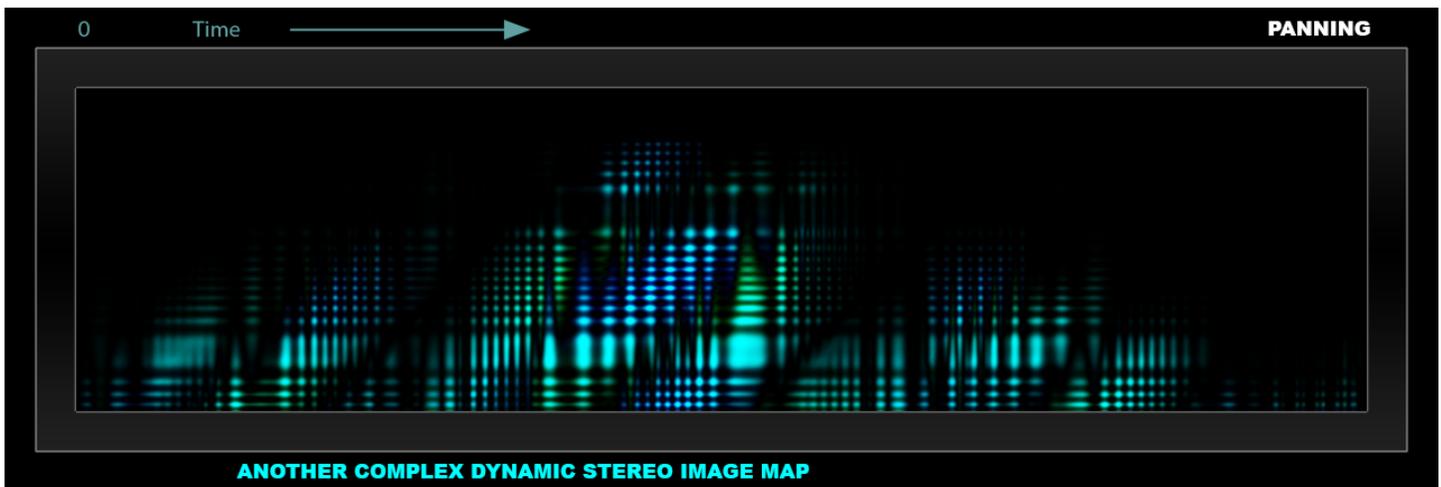
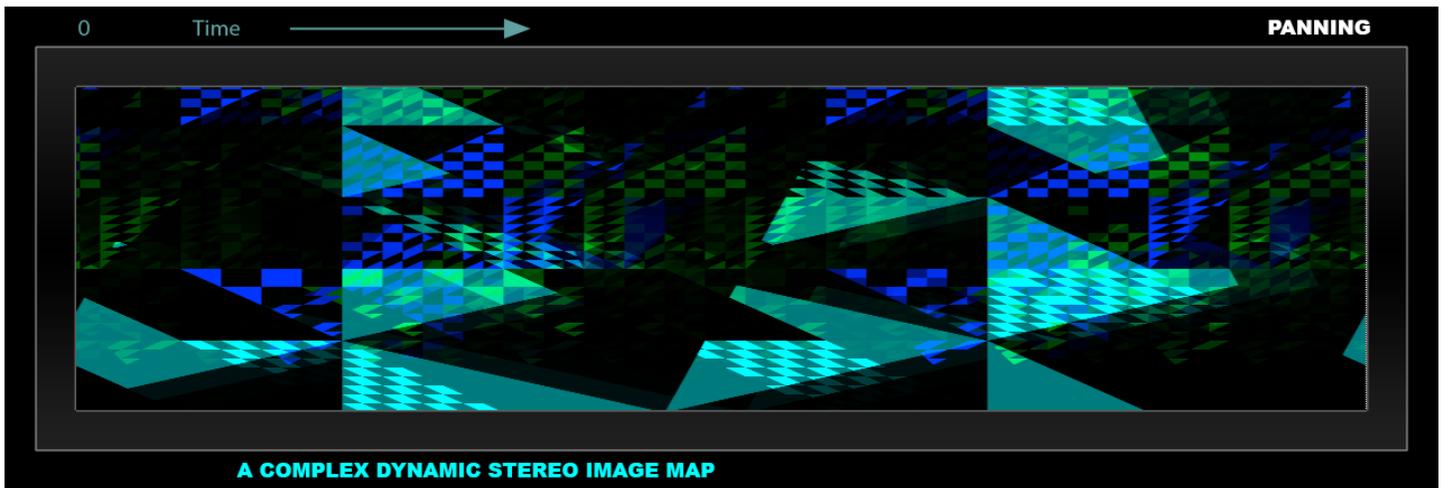


Image Map Application in Kaleidoscope

Now that we understand Kaleidoscope to be a massively parallel bank of resonators and Image Maps have been explained and are understood technically to be a large array of complex envelopes, let us explore how the two work together. Each resonator has its own tuning, input gain, and output gain. Kaleidoscope offers two independent Image Maps: one controls input gain to the resonators, and the other controls output gain from the resonators. Each horizontal line of the two Image Maps represents either the Input or the Output gain envelope for that specific resonator.

- **Input Modulation:** controls the gain of the signal coming into the respective resonator. This controls when and how intensely a particular resonator is excited and caused to oscillate.
- **Output Modulation:** controls the gain of the signal coming out of the respective resonator.

If Feedback is very low these two processes are very similar. If Feedback is exactly zero, and when using FIR mode – a no Feedback mode, the result is indeed **exactly** the same. However when feedback is sufficiently high, the result is not the same due to the “decay tail” of each resonator once it has been excited. Recall that Spring resonators produce exponentially decaying sine waves, and String resonators produce exponentially decaying delays. Once some form of input signal has excited them, they will continue to oscillate or ring even after the input has stopped. Input Modulation is powerless to do anything about this. Output Modulation can be used stop these naturally decaying tails.

If we compare to a piano, Input Modulation allows you to control which keys get struck. Output Modulation is like a per-key sustain pedal that is on by default, but which can be used to abruptly turn off sustaining resonators as desired. In general, Input Modulation is a little more natural sounding. Output modulation is more extreme especially at fast modulation rates. At fast rates and when using rhythmic Image Maps it can produce rhythmic gating effects. At slow rates with smoothly changing Image Maps, it can be used to create slowly evolving variations of the phrase created by input modulation.

Resonance and feedback are more or less synonymous. In order to achieve high resonance, and therefore narrow, highly selective spectral peaks, feedback must generally be high. This means if it is desired to have a very precise tonality in a given preset, feedback must be sufficiently high, and as a consequence there will be a significant decay time. *The 50% value of Kaleidoscope's Feedback knob produces a decay time of one second at the reference frequency for example. Output modulation can be used to "bend the laws of physics" and obtain both high frequency selectivity as well as rhythmically tight modulation.*

Finally, Input Modulation and Output Modulation can have independent modulate rates/periods. This allows for tremendously complex and evolving patterns to be created with relative ease that would take forever to attempt to program by other methods.

Why is Tuning Important?

As we now understand, Kaleidoscope offers up to 512 active resonators, each of which controlled by two independent envelopes. So what do we do with all of these?

Musical Tunings

As mentioned, tuning of each resonator is independent and can be tuned with scientific precision (64-bit floating point) to any imaginable tuning. We could, and do, easily tune the resonators to a standard 12-Tone Equal Temperament scale (i.e. Semitones) and offer a full eight-octave range such a Bösendorfer Model 290 Imperial™ grand piano. We could, and do, offer different "stretch tuning" variations on this standard tuning. We could, and do, offer various historical tunings of the 12-Tone scale such as Pythagorean, Just Intonation, Well Tempered, etc.

Additionally if such tunings are used and "traditional" music structure is the goal, the Image Maps could look something quite similar to, if not identical to, Piano Roll notation in traditional sequencers. This would be a perfectly wonderful way to work with Kaleidoscope, and indeed in a future update we intend to offer MIDI import so that MIDI data can be converted to Image Maps to make this method much easier. In such cases you could feed Kaleidoscope a drum loop or use the built-in white-noise generator, and output a Bach fugue for example. In such cases we would not really need to use all 512 possible resonators. 128, 96, or even 64 lines would be enough.

A large part of the fun of Kaleidoscope, however, lies in the ability to use images that are both aesthetically pleasing visually as well as musically useful. This creates a certain challenge, as real music is generally performed by human beings with two arms, two hands, and ten fingers and thus playing more than ten notes at a given time is quite difficult. Therefore musical performance data is usually comparatively sparse in terms of its vertical density. We don't generally find clusters of 64 different notes being played at the same time by pianists for example. Pianists don't usually play with their forearms, nor would most people find the sound of an elephant sitting on a piano keyboard particularly musical.

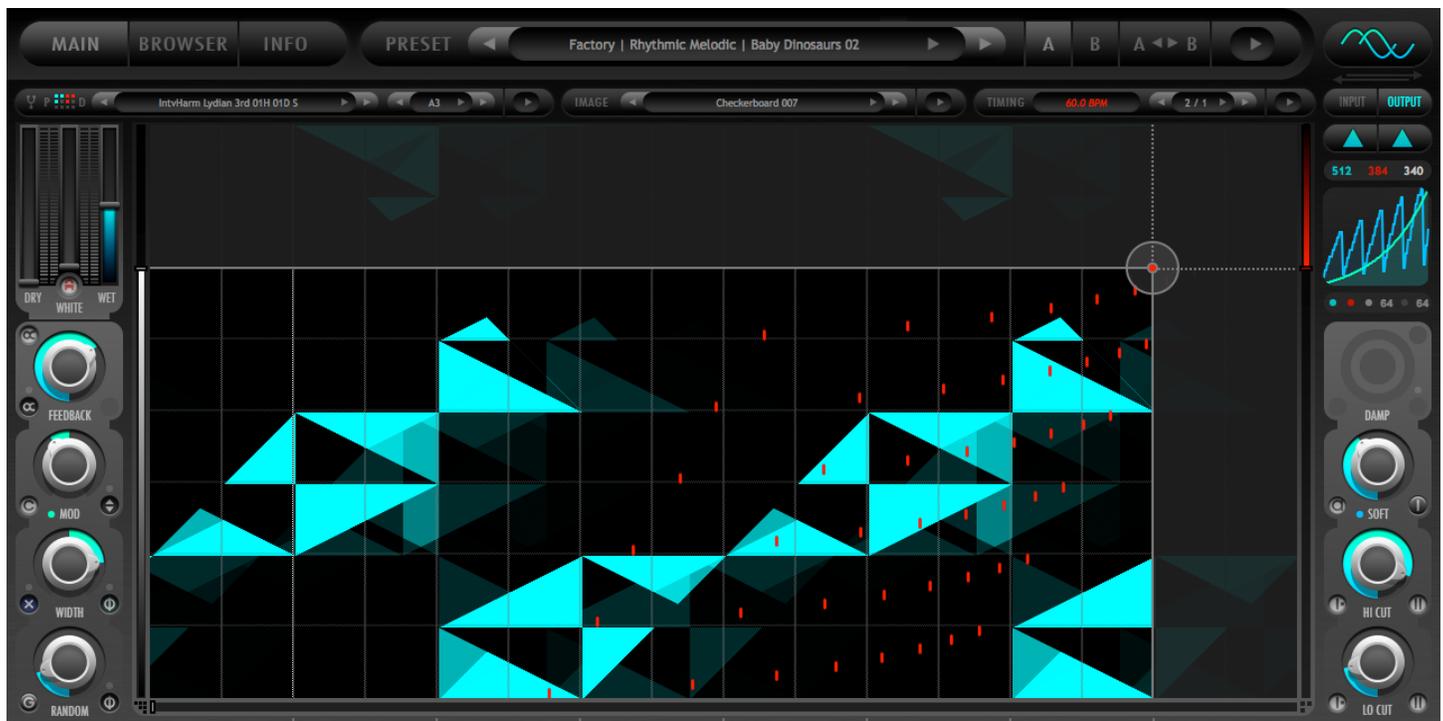
Visual art on the other hand tends to have larger blocks of color, and can be "dense" in both the X and Y-axis. In Kaleidoscope terms, this would translate to many simultaneous resonators being turned on at the same time. Therefore the question becomes, how can we force such states to be musically useful? The answer is we limit the set of available notes to choose from in the tuning, so that there is less chance of displeasing dissonances. The average person is not a huge fan of 12-tone serial music – no offence to Schoenberg or Scriabin – and most people find more simple harmonies much more palatable. Most popular contemporary music is highly tonal and in a single key signature or mode. Therefore

we could, and do, offer tunings that limit the available number distinct musical notes that are possible to be excited. We offer things such as:

- 7 Tone scales and modes
- 5 Tone pentatonic scales
- 3, 4, 5 and more note chords
- Various harmonic series tunings
- Non-Western scales from ethnomusicology

If we reduce the number of available pitches however, we are now using even fewer resonators, and so the question becomes why do we need 512 lines? The answer is we can use two things to increase the number of used resonators:

- **Partials:** provide up to 64 partials (63 Harmonics plus the fundamental) of the given note/frequency
- **Duplicates:** provide up to 64 copies of the same note/frequency



Spring resonators as you recall, do not have harmonics. The Partials feature can be used to add them and the Image Maps can control dynamic gain and panning of each one independently, unlike simply using Strings without adding extra partials. Duplicates can be used with the Random detune knob to create thick choruses of unison notes that can also be controlled dynamically with the Image Maps.

Using Musical (Note) Tuning Modes and Tuning Scales based on musical scales or chords, together with variable Partials and Duplicates can easily generate results that perfectly match the tonality of your musical project while allowing you to use dense, visually pleasing images.

Sound-Design Tunings

Additionally, and perhaps even more importantly to sound-designers, Kaleidoscope is not only concerned with “musical” sound. It is concerned with *all* sound. *Traditionally musical organization of sound is a very small subset of all sound that is possible.* Every human being on earth, as well as every other creature, intelligent or not, here and elsewhere, is exposed on a daily basis to audio signals that follow much different organization principals. Animal sounds, nature sounds, mechanical-industrial sounds, sounds from deep space, music of the spheres, galactic radiation, alien sounds (?)

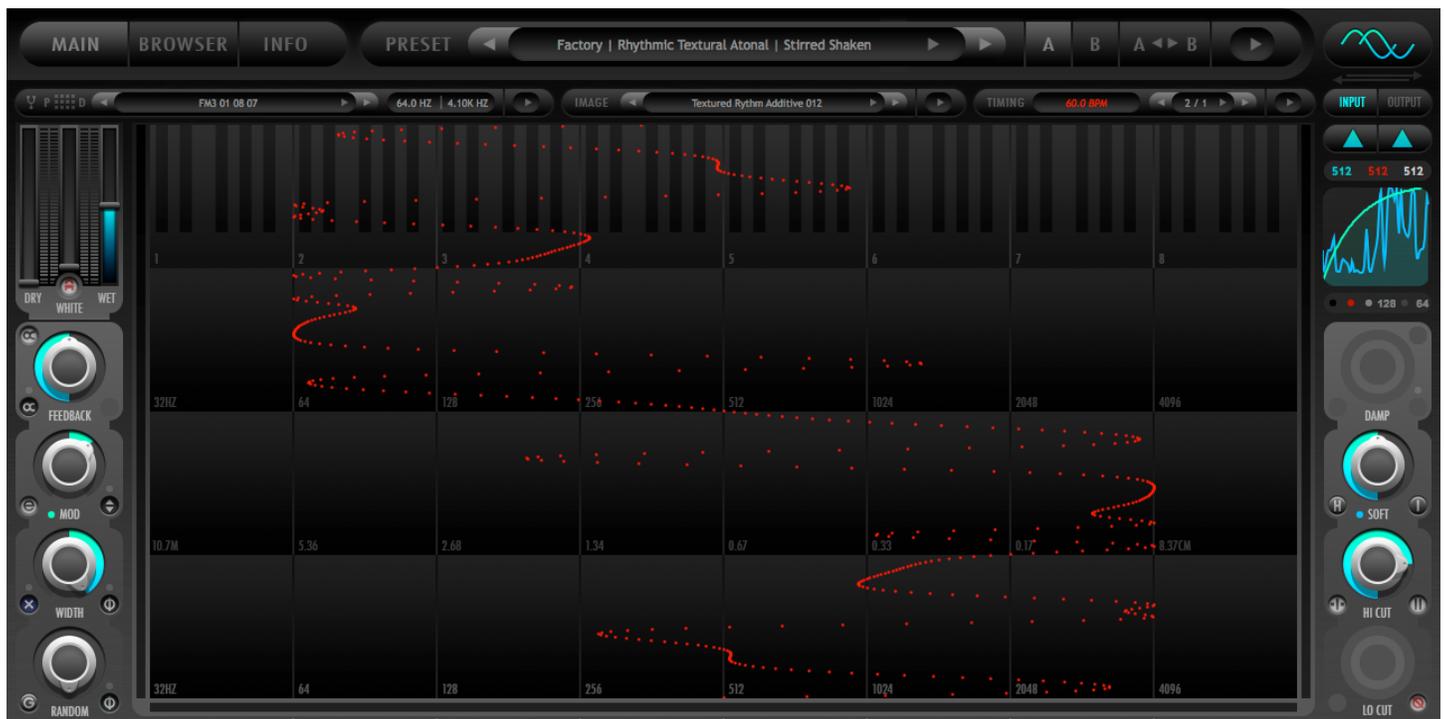
etc. all have their own organizational principles. The good news is that with the right tuning and the right Image Map performance it is possible to emulate such sounds, as well as create an entirely new universe of sounds that have never before been heard – at least by us humans.

All sound in the universe is created by one form of resonance or another. Everything in nature can be modeled through the summation of sine waves and exponentials. Kaleidoscope's Spring resonators produce exponentially decaying sine waves. Summing them together at different frequencies and time varying amounts can, in theory, effectively model anything and everything.

In addition to allowing the reference frequency to be set as a musical note and using standard musical tunings in the form of scale files Kaleidoscope can also set the tuning of each resonator in:

- Absolute Hertz values (useful to match exactly measured resonances of real physical objects or “healing music”)
- Period in Seconds
- Size in Meters (useful to model resonances of large acoustic spaces to create reverb presets using String modes)
- BPM and Sync Period (useful to create tempo synced delay effects using String and FIR modes)

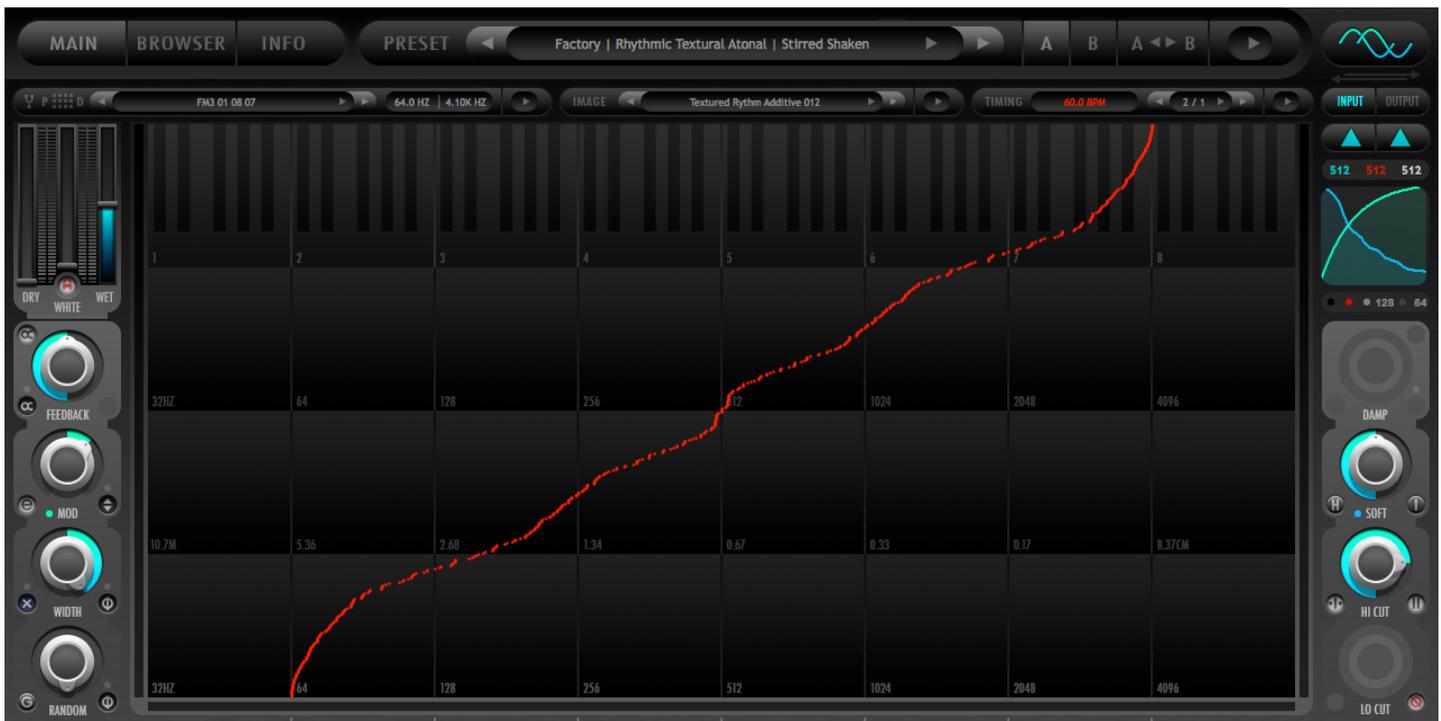
In effort to completely blow your mind, we also allow the use of **Waveforms** to function as tuning maps:



Galbanum Architecture Waveforms for example, can be used in place of Scale files as tuning functions. We offer two bipolar waveform modes where the waveform function is centered on a reference note or frequency, which are good for creating microtonal clusters and spectrums. We also offer a Min and Max Hertz mode where the waveform tuning function is scaled to fit perfectly within the specified hertz range. You could for example, decide you needed some additional textural special effect in your mostly complete, dense electronic music mix. You could look at a spectrum analyzer and find some area of lower energy and use the Waveform Min Max Hertz mode to confine your new sound effect only to this range. Alternatively you might need some eerie high frequency texture effect to wind up the tension and cue impending doom in the sci-fi movie or game you are doing sound-design for. Or perhaps some textural sub-bass drone to emulate the sound of space ship warp drives in homage to Ridley Scott?

Stop wasting time searching through sample libraries for the perfect sound effect that fits your production needs. Instead simply generate a unique one that no one else has completely from scratch according to your exact and specific needs. Waveform tuning modes are excellent for creating experimental atonal special effects!

As you can see above, unlike FFT-based solutions, tunings do not need to be in order. Any resonator line can have any perfectly precise tuning ratio. Gaps between the frequencies of two adjacent resonators can be completely arbitrary and variable. Out-of-order tunings, such as occurs when adding Partials, can produce quite interesting results even with simple Image Maps. A simple diagonal gradient Image Map will give a filter sweep effect for example if the tuning is in-order, but the result will be much more complex and interesting if the tuning is out-of order. Kaleidoscope can optionally sort a given tuning so that the frequency of each resonator line is either in ascending or descending order. This is useful to try different variations of the same spectral content. Additionally, in-order tunings are a bit easier to think about and intuitively grasp the impact of the Image Map, which can be helpful when first learning Kaleidoscope.



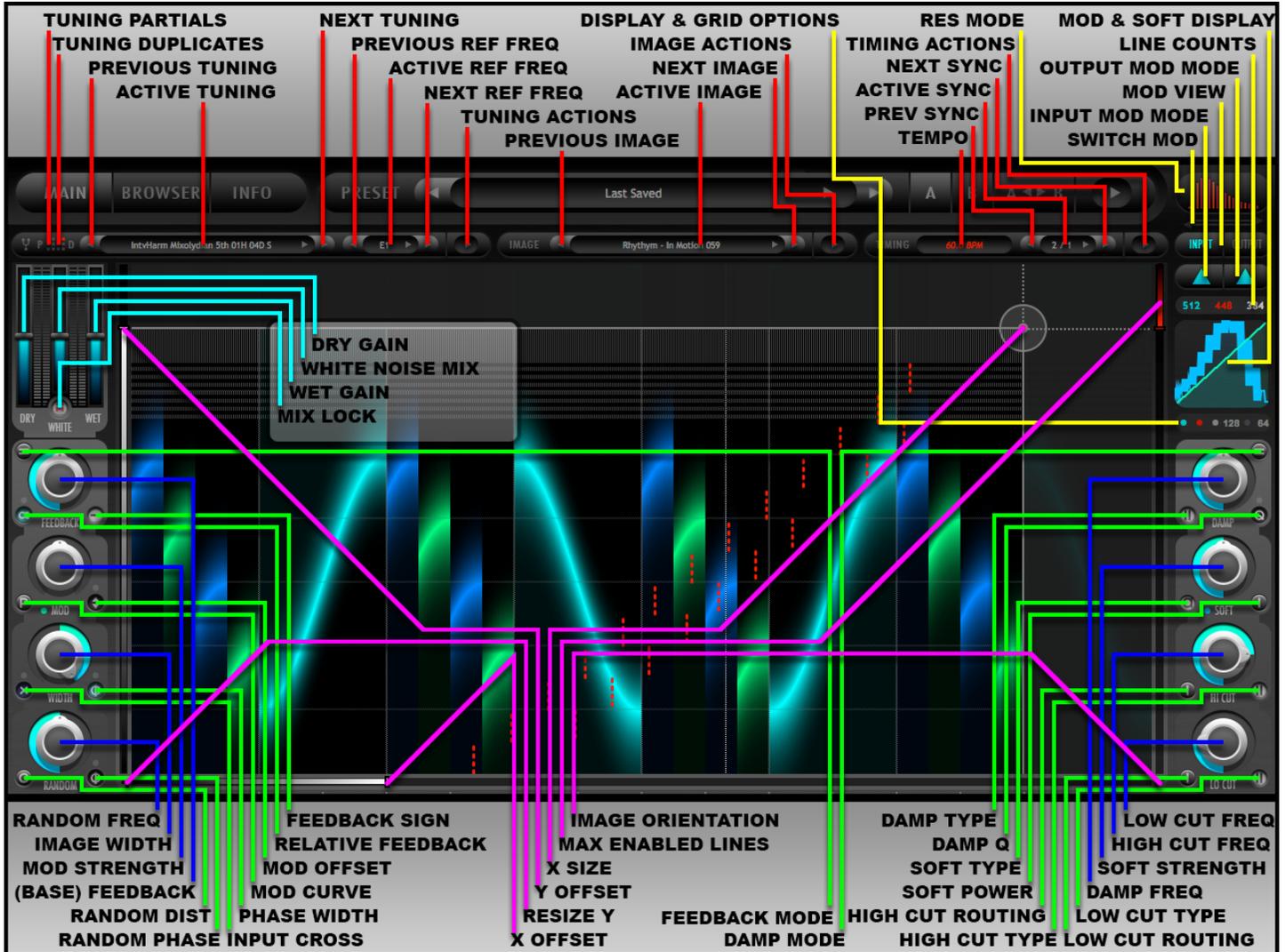
Kaleidoscope displays the tuning function as a series of pixel dots drawn in the unused color channel (i.e. Red, for the Cyan color scheme). Sometimes this function will look fairly smooth. Other times it will look quite messy and chaotic. This is dependent completely on the tuning scale or waveform file used and is an accurate representation of the current tuning.

The tuning function display can be optionally drawn over top of the Image Map so that you can always see a visual representation of the current tuning. Additionally, if the Image Map view is disabled, Kaleidoscope displays a special background that shows various frequency scales including Hertz, Meters, and Musical notes. Each resonator line has a one-pixel dot in it. The horizontal axis is frequency, and the vertical axis is simply an indication of what resonator line it is. The first resonator is at the bottom of the image. The last resonator is at the top of the image. The display range is either 8 or 16 octaves and markers are drawn on the lower edge of the GUI to communicate the current zoom level when the Image Map covers the special tuning background.

Main Page Parameters & GUI Details

Overview

Kaleidoscope's Main page consolidates significant complexity in a reasonably simple manner and displays approximately 70 parameter controls and GUI elements. Details are shown below.



Sub-Menu Area

Kaleidoscope's Sub-Menu Area holds three very important groups of controls: Tuning, Image, and Timing. Each group also has it's own Action Menu Panel which offers additional actions, options, mode switches, and controls.

- **The Tuning Area** contains items that control the currently active Tuning Resource File, the Reference Note or frequency/frequencies, Partials, Duplicates, and the Tuning Action Panel Button.
- **The Image Area** contains items that control the currently active Image resource file and the Image Action Panel Button.
- **The Timing Area** contains items that control the speed of how quickly the Image Map is scanned left to right. It also contains the Timing Action Panel Button.

Both the Tuning Area, as well as the Timing Area have different modes of operation that can be switched in the associated Action Panels. The exact controls that are displayed in each Area are dependent on the current mode.

Partials

Partials control the number of harmonics that are added to the current tuning. Harmonics are integer multiples of the fundamental (i.e. 2, 3, 4, 5, 6....). Spring resonators do not contain harmonics. The Partials function can add them, and dedicate an independent line in the Image Map to each one to give the ultimate control. Using Partials is a quick and easy way to get different variations of a spectrum while maintaining its "musical identity". It effectively changes the timbre of a preset without changing the musical tonality. Using Partials with Strings is not generally necessary, but it does not hurt to try and see if the result is musically useful or not in the context of your project. You should learn to love experimentation.

Partials can have a value of 1, 2, 4, 8, 16, 32 or 64. When Partials is 1, no extra harmonics are added and only the fundamental remains. If Partials is 2, the second harmonic is added to the fundamental. If Partials is 4, the second, third, and forth harmonic are added to the fundamental. Thus the Partial number represents the highest harmonic that is added.

Partials are relative to the existing tuning ratios in the current Scale or Waveform tuning. Each existing ratio in the current tuning will be duplicated the same number of times as the Partials setting, and then will be multiplied by the harmonic number. Adding Partials "expands" the current scale upwards when using Scale tuning. The number of Defined Lines is also multiplied by the Partials value.

If for example the current Scale Tuning used the following four ratios:

{1.0, 1.1, 1.2, 1.3}

It would become the following 16 ratios, in this order, if Partials is set to 4:

{1.0, 2.0, 3.0, 4.0, 1.1, 2.2, 3.3, 4.4, 1.2, 2.4, 3.6, 4.8, 1.3, 2.6, 3.9, 5.2}

When using Waveform Tuning Mode, Partials are added in a way that maintains the overall "geometry" of the waveform as best as possible, instead of expanding the waveform upwards as is the case with Scale Tuning Mode. These topics are best understood by loading a Scale tuning such as Semitones and changing the state of the Partials button while watching the Tuning Display, and then doing the same using a Waveform Tuning for comparison.

Using Harmonics results in tuning functions that are out-of-order as can easily be seen in the Tuning Display. The Tuning Sort Function can be used to keep the frequency space in-order if desired. Both Sorted and Unsorted methods are useful in different circumstances.

Duplicates

Duplicates control the number of duplicates that are added to the current tuning. Duplicates are simply repetitions of the same tuning ratio for multiple resonator lines. Kaleidoscope offers a Random tuning control that allows resonators to be detuned randomly within a user-controlled range. Using Duplicates together with Random detuning can create pleasing chorus-like effects. Indeed many musical instruments in the real world use multiple Strings for the same note for this reason. Slightly detuned unisons generally sound a bit more natural than mathematically perfect single-frequency resonators.

Duplicates can have a value of 1, 2, 4, 8, 16, 32 or 64. Adding Duplicates “expands” the current scale upwards when using Scale tuning. The number of Defined Lines is also multiplied by the Duplicates value. Duplicates are applied after Partials.

If for example the current tuning used the following four ratios: {1.0, 1.1, 1.2, 1.3}

It would become the following 16 ratios if Duplicates is set to 4:

{1.0, 1.0, 1.0, 1.0, 1.1, 1.1, 1.1, 1.1, 1.2, 1.2, 1.2, 1.2, 1.3, 1.3, 1.3, 1.3}

If Partials is also set to 4 the tuning would become the following 64 ratios if Duplicates is set to 4:

{1.0, 1.0, 1.0, 1.0, 2.0, 2.0, 2.0, 2.0, 3.0, 3.0, 3.0, 3.0, 4.0, 4.0, 4.0, 4.0,
1.1, 1.1, 1.1, 1.1, 2.2, 2.2, 2.2, 2.2, 3.3, 3.3, 3.3, 3.3, 4.4, 4.4, 4.4, 4.4,
1.2, 1.2, 1.2, 1.2, 2.4, 2.4, 2.4, 2.4, 3.6, 3.6, 3.6, 3.6, 4.8, 4.8, 4.8, 4.8,
1.3, 1.3, 1.3, 1.3, 2.6, 2.6, 2.6, 2.6, 3.9, 3.9, 3.9, 3.9, 5.2, 5.2, 5.2, 5.2}

Notice that in this case you would effectively have 16 resonator lines dedicated to the same “musical note”.

When using Waveform Tuning Mode, Duplicates are added in a way that maintains the overall “geometry” of the waveform as best as possible, instead of expanding the waveform upwards as is the case with Scale Tuning Mode. These topics are best understood by loading a Scale tuning such as Semitones, and changing the state of the Duplicates button while watching the Tuning Display, and then doing the same using a Waveform Tuning for comparison.

Active Tuning

The Active Tuning Controls display the current tuning and allow you to change it.

- **Active Tuning:** Displays the currently loaded Tuning Resource File. This resource may be a text-based Scale file, or a Waveform (64-bit, Wav format) depending on the current Tuning Mode. Clicking anywhere in the display field will open a Resource Library Browser Panel for currently used resource type.

**See further details in the “Resource Library Browser Panel” section.
See further details on Tuning File Formats in the Appendix.**

- **Previous Tuning:** Loads the tuning resource file (Scale or Waveform) that comes alphabetically before the currently selected tuning resource file.
- **Next Tuning:** Loads the tuning resource file (Scale or Waveform) that comes alphabetically after the currently selected tuning resource file.

Tuning Reference

Tuning Scale and Waveform files provide relative tuning ratios, not absolute values in Hertz. They are saved simply as a list of double precision floating point numbers and do not have any innate measurement units. In order to convert the

relative ratios to specific Hertz values, a Tuning Reference must be given. The Tuning Ratios are multiplied by the Tuning Reference to obtain Hertz values for each line.

It is possible to set the Tuning Reference exactly in Hertz if desired; for example, if the current tuning used the following four ratios: {1.0, 1.1, 1.2, 1.3}

And the Tuning Reference was set to 100 Hertz, the actual tuning of each line in Hertz would simply be:

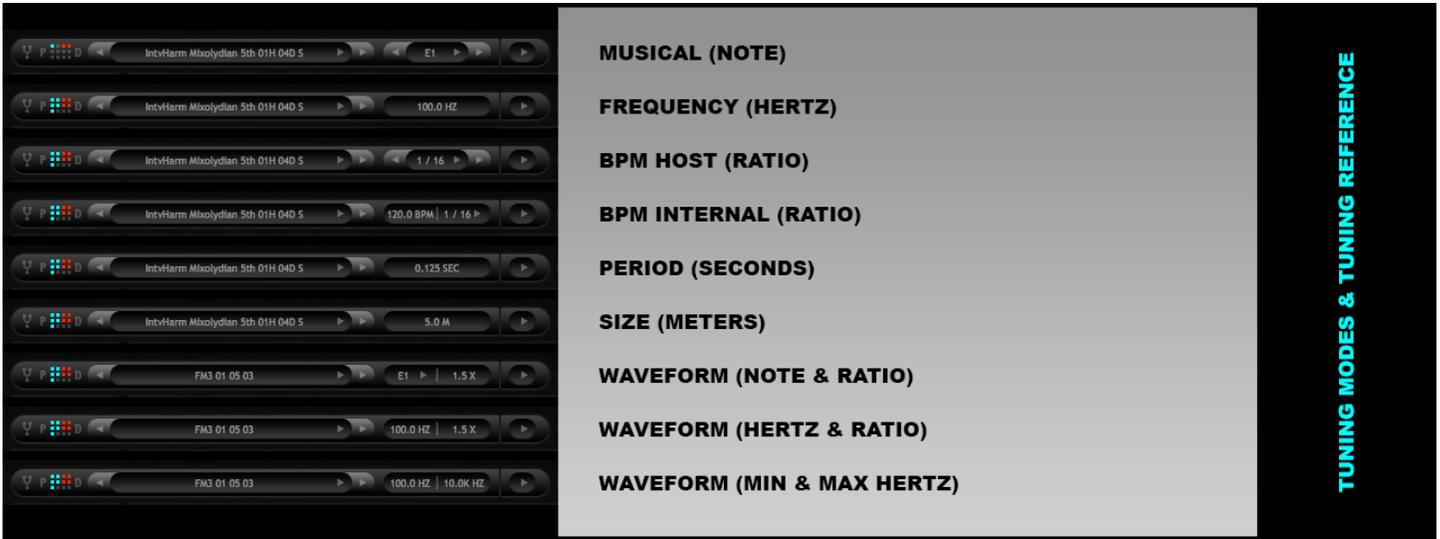
$$\{1.0 * 100, 1.1 * 100, 1.2 * 100, 1.3 * 100\} = \{100, 110, 120, 130\}$$

This is one method of setting the Tuning Reference. The Tuning Reference can actually be set in nine different ways depending on the current Tuning Mode. All of the following Tuning Modes are used in conjunction with Tuning Scale Resource Files (.txt). The Tuning Reference control area looks slightly different depending on the current Tuning Mode:

- **Musical (Note):** The Tuning Reference is set by selecting a musical note (A, A#, B, C...) and an octave. A preference on the Info page of the GUI establishes a global reference for this mode (i.e. A = 440 Hz), and all other Musical Note values are derived from this global preference assuming standard 12-Tone Equal Temperament. This is sufficient for 99% of most musical-usage needs.
- **Frequency (Hertz):** The Tuning Reference is text-entered in exact full double-precision Hertz as desired.
- **BPM Host (Ratio):** The Tuning Reference is set as a function of Host Tempo and a fractional Sync Period. This is useful to create tempo synced delay effects when using String or FIR resonator modes. Host Tempo is used.
- **BPM Internal (Ratio):** The same as the BPM Host method, except that Tempo is manually text-entered.
- **Period (Seconds):** The Tuning Reference is text-entered in exact full double-precision Seconds as desired.
- **Size (Meters):** The Tuning Reference is text-entered in exact full double-precision Seconds as desired.

The final three Tuning Modes use Tuning Waveform Resource Files (.wav). These modes use *two* Tuning Reference values.

- **Waveform (Note & Ratio):** A musical note is selected, and a Ratio is text-entered that scales the ratios in the Waveform Tuning file in a bipolar range centered on the note value. This is useful for creating microtonal clusters around specific notes.
- **Waveform (Hertz & Ratio):** A Hertz value is text-entered, and a Ratio is text-entered that scales the ratios in the Waveform Tuning file in a bipolar range centered on the Hertz value. This is useful for creating microtonal clusters around specific frequencies.
- **Waveform (Min & Max Hertz):** Both a minimum and a maximum value in Hertz are text-entered and the ratios in the Waveform Tuning file are scaled to fit this range.



TUNING MODES & TUNING REFERENCE

Tip: Use “Frequency (Hertz)” Tuning Mode, and a Reference Frequency of 1.0 Hertz to explicitly set resonator lines in exact Hertz.

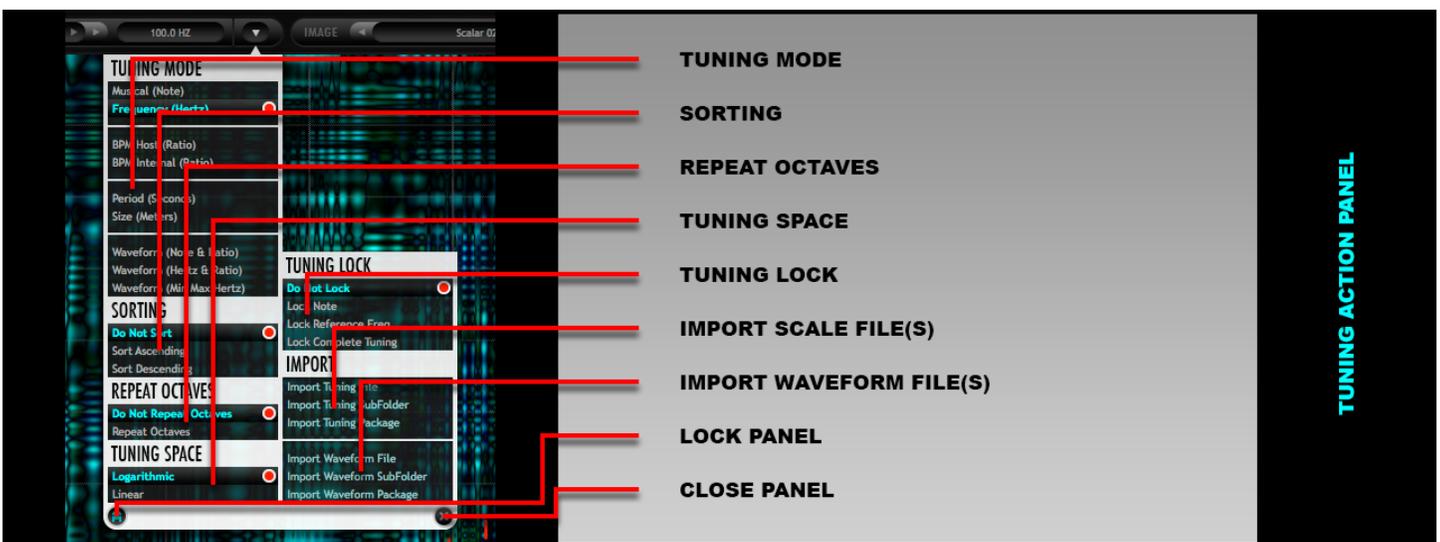
Tuning Action Menu Button

The Tuning Action Menu Button opens the Tuning Action Panel.

Tuning Action Panel

Action Panels are a form of custom menu that hold macro settings such as Mode changes, as well as actions, extra options, and settings. **All panels have Lock and Close buttons on them. If the Lock button is on, the panel will stay open until clicking on the close button closes it. If the Lock button is off, the panel will close automatically when a selection is made.**

The Tuning Action Panel displays several items of this nature all related to Tuning:



TUNING ACTION PANEL

Tuning Mode

The Tuning Mode Option List allows you to select which Tuning Mode Kaleidoscope is currently using. Tuning Mode selection effects whether Kaleidoscope uses Scale Files or Waveform Resource Files to derive tuning ratios, as well as

how the Tuning Reference is set. These two things combined with the currently selected Tuning Resource File, determine the exact tuning of each resonator. The Tuning Modes were previously discussed above.

Option Lists are an enumerated list of all of the options a particular control can have. Only one option may be selected at a given time. Color highlights, a 3D selected-item box, and a circular marker denote the currently selected option.

Sorting

Sometimes Tuning Ratios are out-of-order. Adding Partials to the current tuning can create this situation for example, and many Waveform Tunings are not in order. This is perfectly acceptable, however it may be desirable to sort the Tuning Ratios. Sorted tunings make it easier to gain an intuitive understanding of how a particular Image Map will sound. Sorted tunings make it easy to create filter sweep effects of any arbitrary spectrum. Unsorted tunings can be interesting in that they tend to keep the “identity” of clusters of particular lines that contain harmonics more intact. For example if 64 lines on an Image Map all represent the same musical note due to the user of Partials and Duplicates, it can be interesting to leave the tuning unsorted because this is more likely to give the perception that these lines are functioning together as a single note – assuming the Image Map is smoothly changing in the Y-axis. There are no hard rules regarding sorting.

- **Do Not Sort:** should be self-evident
- **Sort Ascending:** should be self-evident – lowest frequency is on the bottom of Image Map
- **Sort Descending:** should be self-evident – highest frequency is on the bottom of Image Map

The state of the Sorting parameter is persistent across changing other Tuning settings including changing Tuning Resource File, Changing Tuning Modes, and everything else. This is quite useful if you know that you want to work specifically one of these options.

Repeat Octaves

Some Tuning Scales do not contain a large number of **Defined Lines**. Tuning Scale Files use the “Scala” file format syntax originally created by Manuel Op de Coul in the Netherlands. This format has become somewhat of a standard for exploring alternative tunings. In this format it is common to assume the scale repeats every octave. Therefore when creating a Natural Minor scale for example, only 7 Tuning Ratios have to be defined, and these are assumed to repeat every octave.

In Kaleidoscope, sometimes this behavior is desirable, other times it is not; therefore this option is offered to control whether or not Tunings should repeat every octave. This option only has an effect when the product of the number of Scale Lines, Partials, and Duplicates is less than the height of the Image Map (512) (i.e. you are not already using the maximum number of Defined Lines). It is most common to have a need for this option when dealing with musical tunings such as musical scale (major, minor, etc.) and chord-based Tuning Scale Files. This option has no effect when using Waveform Tunings since Waveform Tunings already provide more than 512 possible Tuning Ratios.

Tuning Space

Tuning Space affects the frequency mapping of Waveform Tunings. When using Waveform Tuning Modes, the given Waveform is normalized to fit the endpoints of the specified frequency range. The rest of the points are mapped evenly within this range. “Evenly” can be somewhat ambiguous however, and can be done in two ways:

- **Logarithmic:** an equal factor is **multiplied** each sampling step
For example 12-Tone Equal Temperament is Logarithmic: $y(n) = 2^{(n/12)}$
 $y(1) = 2^{(1/12)} \approx 1.059$
 $y(2) = 1.059 * 1.059$
 $y(3) = 1.059 * 1.059 * 1.059$

- **Linear:** an equal factor is **added** each sampling step
For example the Harmonic Series is Linear: $y(n) = n$
 $y(1) = 1$
 $y(2) = 1 + 1$
 $y(3) = 1 + 1 + 1$

Musical Notes and intervals are on a logarithmic scale for example, but harmonics are on a linear scale. Therefore we allow both options. Logarithmic tends to subjectively sound a little more balanced, and linear tends to have sonic characteristics of a harmonic series and results in more energy in higher frequencies. Both options are useful.

Ironically, just in case your head has not exploded yet, the Tuning Display is itself on a logarithmic scale, as almost all audio-industry frequency scales are, so if you use a waveform such as normal saw/ramp function, which is itself linear in the WAV file, it will be displayed as a straight line when using the Logarithmic Mode, not the Linear mode. Basically under most circumstances you want to be able to map a waveform in a linear fashion in a Logarithmic tuning space, therefore Logarithmic mode is usually the desired choice. If you followed that explanation perfectly, congratulate yourself!

Usually you want Logarithmic, but try both. There are no rules.

Tuning Lock

Tuning Lock affects the behavior of the various Tuning settings when changing presets. If you are using Kaleidoscope for musical purposes it is commonly expected that the project you are working on is in a particular musical key signature. You may therefore know that you need a preset that matches your key signature by choosing an appropriate scale, chord, or at least tonic note. If you are working in C-Minor for example, auditioning a preset in C#-Major is pretty certain to sound like a train-wreck. Therefore we allow you to lock the current state of the tuning settings in different ways so that it is easier to find presets that work well with the tonality of your current music project.

- **Do Not Lock:** Nothing is locked. Presets will load exactly as saved.
- **Lock Note:** Locks only the Note value. Does not lock the Octave. Does not lock Hertz Tuning Reference values. Does not lock the Scale. Does not lock the Waveform. Does not lock the Tuning Mode. This is useful if you just want to match the tonic, but preserve the frequency range of the preset reasonably well (i.e. a bass preset will not become a soprano preset). Atonal Waveform-based FX presets are unaffected.
- **Lock Reference Freq:** Locks the Note value. Locks the Octave. Locks Hertz Tuning Reference values. Does not lock the Scale. Does not lock the Waveform. Does not lock the Tuning Mode. This is useful if you just want to match the tonic, and the frequency range of the preset reasonably well (i.e. you want a bass preset and don't care if the original preset was designed to be a soprano preset). Atonal presets will still switch to being atonal, but frequency ranges in the presets will be ignored.
- **Lock Complete Tuning:** Locks everything. You can preserve an exact musical Scale and Reference. You can preserve an exact atonal Waveform spectrum. All tuning related settings are locked. This can give unpredictable results if you are loading atonal presets locked to load with a musical scale or vice versa, but maybe a happy accident is just what your project needs?

Since the Tuning Lock feature is quite powerful and convenient, most of the musical factory presets are set simply to use "A" as a reference pitch. It is expected that users will change and lock this to their specific project needs.

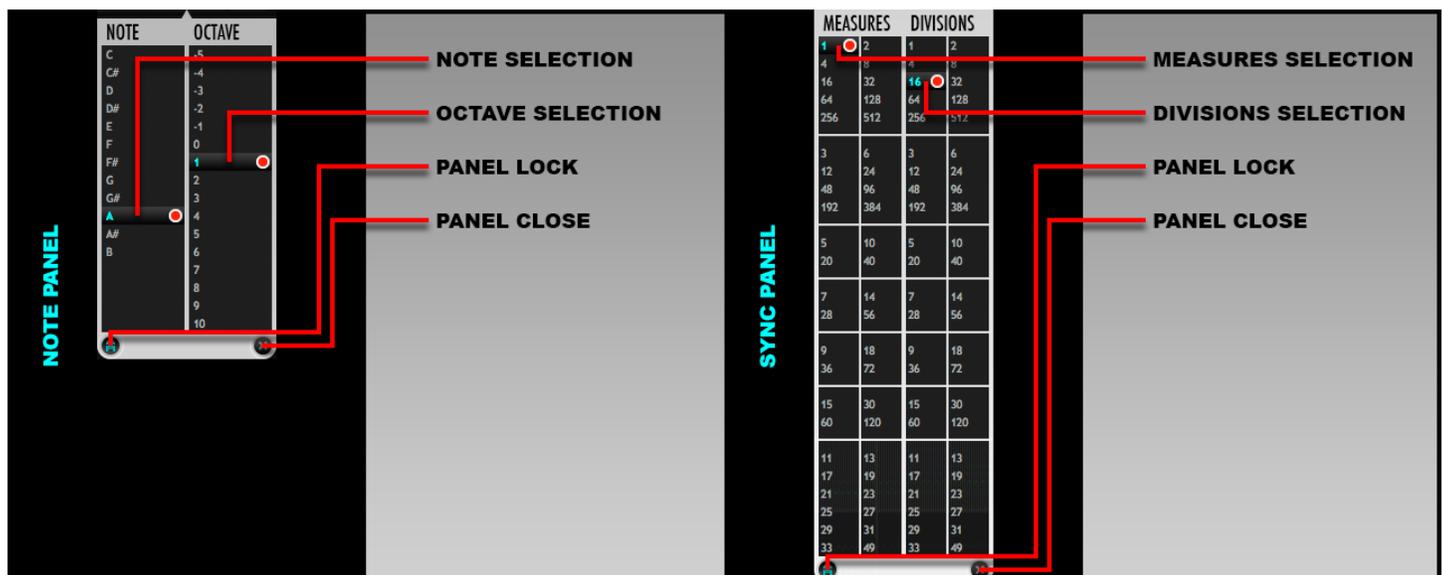
Tuning Import Actions

Tuning Import Actions allow you to add more Tuning Resource Files to your 2CAudio Resource Library and make them immediately available for use without reloading Kaleidoscope.

- **Import Tuning File:** Imports a single Scale file to
`\2CAudio Resource Library\Tuning Library\User\My Tunings\`

- **Import Tuning SubFolder:** Imports a single Scale folder to
 $\backslash 2\text{CAudio Resource Library}\backslash\text{Tuning Library}\backslash\text{User}\backslash$
- **Import Tuning Package:** Imports a complete Scale directory to
 $\backslash 2\text{CAudio Resource Library}\backslash\text{Tuning Library}\backslash$
- **Import Waveform File:** Imports a single Waveform file to
 $\backslash 2\text{CAudio Resource Library}\backslash\text{Waveform Library}\backslash\text{User}\backslash\text{My Waveforms}\backslash$
- **Import Waveform SubFolder:** Imports a single Waveform folder to
 $\backslash 2\text{CAudio Resource Library}\backslash\text{Waveform Library}\backslash\text{User}\backslash$
- **Import Waveform Package:** Imports a complete Waveform directory to
 $\backslash 2\text{CAudio Resource Library}\backslash\text{Waveform Library}\backslash$

Note and Sync Panels



When using “Musical (Note)” Tuning Mode Kaleidoscope uses a custom Note Panel to make the selection. The left column provides the selection of the Note. The right column allows selection of an Octave. The Octave range is intentionally quite extreme as it may be interesting for sound-design purposes to make the reference frequency in the sub-sonic or ultra-sonic range, and then use various Tuning Ratios in Scales and Waveforms that would bring the resulting frequencies back into the audio range. For example, high harmonics of a very low note, or sub-harmonics of a very high note could be interesting.

Very low frequency (long period) Strings, as well as FIR mode, can be used to create delay effects.

The Sync Panel is used tune the resonators to period lengths that are musical note lengths. Since String resonators produce a series of exponentially decaying delays (with optional frequency dependent damping), if the period of these delays is long enough, the effect will no longer be perceived as frequency effect; it will be perceived as a delay effect. Kaleidoscope allows a **very extreme** range of allowed resonator frequencies/periods. The lowest frequency allowed for

Strings and FIR mode is 0.125 Hertz! This equates to a period of 8 seconds! This translates to 2 Measures at 120bpm. Therefore, we make it possible to set resonator frequency/period in terms of musical note lengths.

Our system of musical “time keeping” is far more advanced than most other products that we are aware of. Most delay effects for example will allow tempo-synched delays to be set as straight-notes, dotted notes, and triplets. We go much, much further. Our sync panel provides a fractional ratio that controls the period length of the delay as a fraction of one measure. The numerator of the ratio is labeled “Measures”. The denominator is labeled “Divisions”. The easiest way to think of this system is:

Every M Measures, there will be D Divisions or repetitions, where M is the number selected for Measures, and D is the number selected for Divisions.

Some examples will help explain this:

- $1/1$ = “Every 1 Measure, there will be 1 Division”. Therefore the result is a whole note period.
- $1/4$ = “Every 1 Measure, there will be 4 Divisions”. Therefore the result is a quarter note period.
- $1/16$ = “Every 1 Measure, there will be 16 Divisions”. Therefore the result is a 16th note period.
- $3/8$ = “Every 3 Measures, there will be 8 Divisions”. Therefore the result is a dotted quarter note.
- $3/16$ = “Every 3 Measures, there will be 16 Divisions”. Therefore the result is a dotted 8th note.

Tuplets are handled like this:

- $1/3$ = “Every 1 Measure, there will be 3 Divisions”. Therefore the result is half-note triplets.
- $1/6$ = “Every 1 Measure, there will be 6 Divisions”. Therefore the result is quarter-note triplets.
- $1/12$ = “Every 1 Measure, there will be 12 Divisions”. Therefore the result is 8th-note triplets.
- $1/24$ = “Every 1 Measure, there will be 24 Divisions”. Therefore the result is 16th-note triplets.
- $1/5$ = “Every 1 Measure, there will be 5 Divisions”. Therefore the result is a quarter-note quintuplet.
- $1/10$ = “Every 1 Measure, there will be 5 Divisions”. Therefore the result is an 8th-note quintuplet.
- $1/7$ = “Every 1 Measure, there will be 7 Divisions”. Therefore the result is a 7th-tuplet.
- $1/14$ = “Every 1 Measure, there will be 7 Divisions”. Therefore the result is a 7th-tuplet that is twice as fast.
- $2/7$ = “Every 2 Measures, there will be 7 Divisions”. Therefore the result is a 7th-tuplet that is twice as slow.

Things can get significantly freakier however:

- $32/25$ = “Every 32 Measures, there will be 25 Divisions”.
- $33/32$ = “Every 33 Measures, there will be 32 Divisions”.
- $384/49$ = “Every 384 Measures, there will be 49 Divisions”.

These sorts of “bizarre” Sync Periods are more useful in the Timing Sync panel that will be discussed shortly. The same Sync Panel design is shared for both Tuning and Timing.

The order of the numbers in the Sync Panel is chosen based on predicted musical usefulness. Numbers are grouped by their base number, and offer various multiples of two for the given base. (i.e. 3, 6, 12, 24, 48, 96...). The musical result is likely to get more complex and strange the lower you go in the Sync Panel groups. It turns out human beings generally prefer simple relationships and simple relationships are the basis for most musical structures. In fact, we surmise 95% of all Western music could be created using only multiples of the numbers 2 and 3. The final group offers various complex numbers designed to create intentionally odd results (pun intended). Again these are more useful in the content of the Timing Sync panel. Pure sound-design applications have far fewer rules however, and “freaky” may just be perfect!

It is important to remember that whatever value is chosen in the Tuning Sync Panel, this value together with the tempo simply determine a Tuning Reference. The actual realized ratios used for each resonator line will be the product of the Tuning Reference, and the Tuning Ratio for the given line.

You could, for example, set each line to 16th note increments by using a Tuning Reference of 1/16, and a Harmonics scale:

{1, 2, 3, 4, 5...}

This would produce ratios of:

{1/16, 2/16, 3/16, 4/16, 5/16...}

Image Maps could then be used to turn these various delay lengths on and off and move them around in spatial position. This can create very interesting and novel delay effects that are unlike typical delays.

Active Image

The Active Image Controls display the current Image Map for the currently viewable Modulation Source and allow you to change it.

- **Active Image:** Displays the currently loaded Image Resource File for the current Modulation view. Kaleidoscope offers independent settings for both Input and Output modulation. The status of the View Switch determines which Image Map is shown in the Active Image display and which will be changed by changes made here. Clicking anywhere in the display field will open a Resource Library Browser Panel for currently used resource type.

*See further details in the “Resource Library Browser Panel” section.
See further details on Image File Formats in the Appendix.*

- **Previous Image:** Loads the Image Resource File that comes alphabetically before the currently selected Image Resource File in the current directory.
- **Next Image:** Loads the Image Resource File that comes alphabetically after the currently selected Image Resource File in the current directory.

Image Action Panel

Action Panels are a form of custom menu that hold macro settings such as Mode changes, as well as actions, extra options, and settings. The Image Action Panel displays several items of this nature all related to Images.

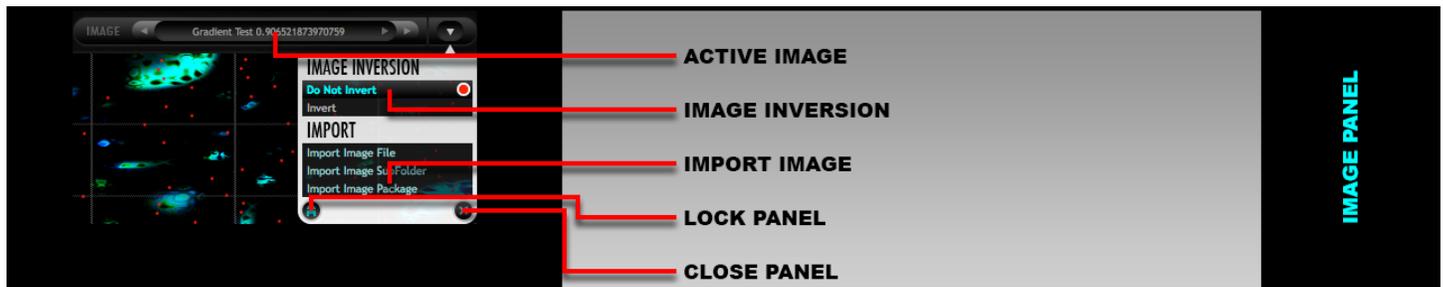


Image Inversion

Image Inversion optionally inverts the brightness of an Image. This has the effect of inverting all resonator modulation envelopes: what used to be full-gain will now be silence and vice versa. Inversion is applied before other image transformations that are possible with other controls on the GUI, therefore inverted images may not look like an exact inversion of the previously image as expected because the inverted image has also been subjected to additional transformations, which themselves are not inverted: namely the Mod Strength control.

- **Do Not Invert:** Loads the Image in the standard manner as found on disk.

- **Invert Image:** Inverts the Image brightness.

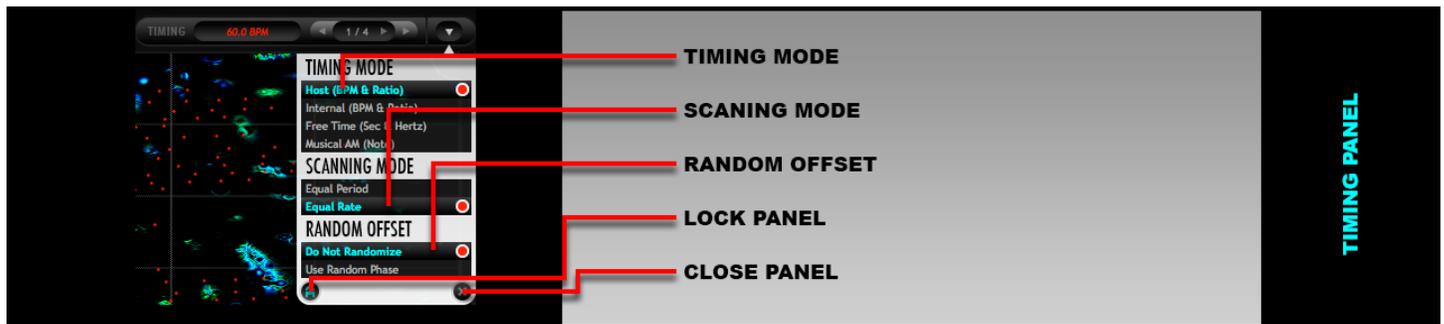
Image Import Actions

Image Import Actions allow you to add more Image Resource Files to your 2CAudio Resource Library and make them immediately available for use without reloading Kaleidoscope.

- **Import Image File:** Imports a single Image file to
 \2CAudio Resource Library\Image Library\User\My Images\
- **Import Image SubFolder:** Imports a single Image folder to
 \2CAudio Resource Library\ Image Library\User\
- **Import Image Package:** Imports a complete Image directory to
 \2CAudio Resource Library\ Image Library\

Timing Action Panel

Action Panels are a form of custom menu that hold macro settings such as Mode changes, as well as actions, extra options, and settings. The Timing Action Panel displays several items of this nature all related to Timing (i.e. how quickly an Image Map is scanned left to right). All items in the Timing Action Panel are independent for Input and Output Modulation.



Timing Mode

Kaleidoscope scans an Image Maps left to right over time (when using Dynamic Modulation Mode). The horizontal axis of an Image Map represents time. The Timing Mode option list determines the rate at which the Image Map is scanned, and thus the length of its period as well (i.e. how long does a single cycle through the Image Map take). There are four different Timing Modes. The two control areas in the Timing controls sub-menu area change depending on the selected Timing Mode:



- **Host (BPM & RATIO):** Tempo information (BPM) is read from the host application and is not editable. The current tempo is displayed in the left control area and is drawn in a colorized italics font to denote that it is not editable. If the host application does not supply tempo information to Kaleidoscope, as is the case with many

stereo editor applications, the Default Tempo Preference on the Info Page of the Kaleidoscope GUI is used. The right control area provides a Sync Ratio, which is changed via the associated Timing Sync Panel. The Sync Ratio consists of a two number ratio that represents “fractions of measures” at the current tempo.

- **Internal (BPM & RATIO):** This is the same as the previous option except that Tempo is manually entered instead of being read from the host. This can be useful if the host has tempo changes (which are not handled smoothly in Kaleidoscope at the moment), or if a tempo other than the one used in the project is desired for some creative reason.
- **Free Time (Hertz & Seconds):** This mode allows the rate to be directly in Hertz, and/or the Period to be set directly in Seconds. Rate is simply the reciprocal of period ($\text{Rate} = 1 / \text{Period}$). Both fields are simply two ways to represent the same value. Adjusting one numeric value will adjust both fields. This mode is useful to match events to specific durations. As an example a sound-designer for visual media might need a sound effect that lasts exactly 5.63 seconds to match the needs of the film/show/game.
- **Musical AM (Hertz & Seconds):** This mode allows the rate to be in set in the Hertz equivalents of Musical Notes. The Note Panel is used to select a Note and an Octave. The Hertz equivalent is shown in a non-editable field on the left. The selected Note together with Global Tuning Reference on the Info Page of the GUI, determines the exact value in Hertz. This mode is mostly used for experimental sound-design and creating intentional distortion effects via creating aliasing by scanning the Image Map at extreme rates.

Scanning Mode

Kaleidoscope has a parameter called X-Size. X-Size allows you to use less than the full Image Map width (i.e. instead of using all 1024 pixels, you could use 960 or 768 etc.) Scanning Mode determines how the scanning rate of an Image Map is affected by changes in X-Size.

- **Equal Period:** This mode preserves the period length established by current Timing Mode regardless of changes to X-Size. As an example, if Timing Mode is Host (BPM & Ratio), Tempo is 120 BPM, and Sync Period is 1/1, the current width of the Image Map (X-Size) will always represent exactly 1 measure at 120 BPM. It does not matter if X-size is 37 pixels or 1024 pixels – both will result in exactly one measure in this example. Therefore scanning rate is adjusted to always fit the current X-Size to the desired period length. This is useful to if you would like to “crop” an Image Map to use only part of it, but do not want to adjust the timing.
- **Equal Rate:** This mode is effectively the opposite of Equal Period. The scanning rate is established assuming a full Image Map width of 1024 pixels. The scanning rate is kept constant regardless of changes to X-Size. This means the length of the period of one cycle of the Image Map is no longer exactly what is shown by the Timing Mode parameters. This mode is useful for two reasons:
 - Rhythmic Image maps that have discrete pulses in them can maintain their rhythmic bases when adjusting X-Size. For example if Sync Period is 1/1, but the Image Map supplies 16 rhythmic pulses, this will create the perception of 16th notes and there will be 16 in each cycle of the Image Map. If the X-Size is set to 768 instead of 1024, it is now 75% of its original size, and this example Image Map now shows 12 pulses instead of 16. If the scanning rate is adjusted to maintain the period length of 1/1, we now have 12 pulses per measure, which is 8th note triplets instead of 16th notes. This is what would happen if Equal Period was used. If Equal Rate is used instead, the pulses would still be 16th notes, but the total length of the pattern would be $\frac{3}{4}$ measures instead of exactly one measure.
 - Due to the change in the effective Period Length caused by Equal Rate, polyrhythms can be established between the Input and Output mod modes with both sharing the same rhythmic base. If both Image Maps have images that supply some form of 16th note pulse for example, and one Image map has an X-Size of 1024, while the other has an X-Size of 768 this establishes a 4/3 polyrhythm while still maintaining the same 16th note base. This can create very complex and intricately evolving rhythmic phrases that have a composite period length of much longer than the two individual Image Map Periods.

If creation of polyrhythms is the goal, changing X-Size and using Equal Rate is the way to do it when the Image Maps have “discrete rhythmic events” in them and some form of embedded pulse that you want to maintain.

If the Image Maps are smoothly changing gradients that do not have “discrete rhythmic events” and are designed to tile perfectly, this method should not be used. In such cases X-size should remain at full width, and different Sync Period values can be used for Input and Output. Many factory presets demonstrate these two concepts.

When X-Size is full width (1024 pixels), Equal Period and Equal Rate are equivalent it does not matter which is selected.

Random Offset

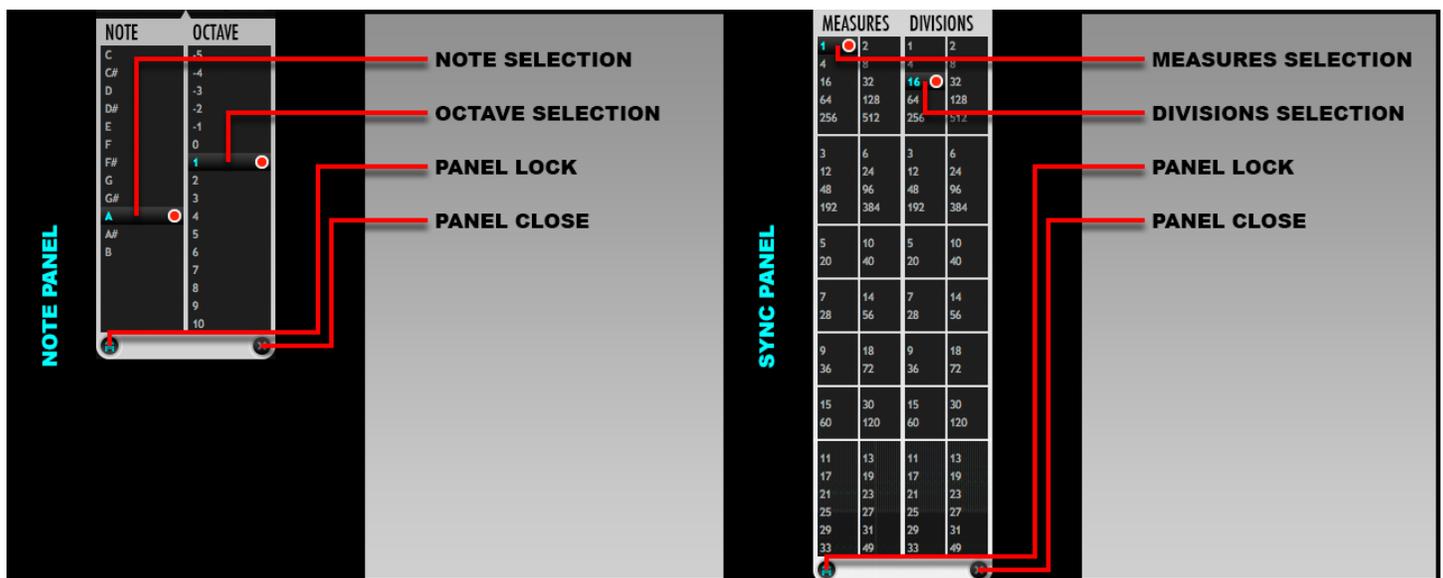
Random Offset optionally subjects the Modulation Envelope for each resonator line to an independent random delay. This can be used to achieve some “humanization” of the timing of modulation envelopes. Additionally it can be useful to keep modulation envelopes in sync with delay tap locations when using very large Random Phase parameter values.

- **Do Not Randomize:** Modulation Envelopes are always all exactly in sync and perfectly phase aligned to the expected position in time.
- **Use Random Phase:** the modulation envelope for each resonator line is subjected to a random delay that is equal to the random delay used by the Random Phase parameter for the current line.

When used with small nominal values of Random Phase, Random Offset is a method to get a sub-pixel timing resolution, in a random manner. When used together with very large Random Phase values it can produce granular-like effects.

Timing Note and Sync Panels

Timing Note and Sync Panels are visually and functionally the same the Tuning Note and Sync Panels. Please see the previous explanation for details regarding the Note Panel.



The Timing Sync Panel is used to set the time period of one cycle of scanning the Image Map to the given fractional ratio of one measure.

Our system of musical “time keeping” is far more advanced than most other products that we are aware of. The easiest way to think of this system is:

Every M Measures, there will be D Divisions or repetitions. Where M is the number selected for Measures, and D is the number selected for Divisions.

Some examples will help explain this:

- $1/1$ = “Every 1 Measure, there will be 1 Division”. Therefore the resulting period is one measure.
- $4/1$ = “Every 4 Measures, there will be 1 Division”. Therefore the resulting period is four measures.
- $256/1$ = “Every 256 Measures, there will be 1 Division”. Therefore the resulting period is 256 measures.
- $1/16$ = “Every 1 Measure, there will be 16 Divisions”. Therefore the resulting period is a 16th note period.
- $3/8$ = “Every 3 Measures, there will be 8 Divisions”. Therefore the resulting period is a dotted quarter note period.
- $3/16$ = “Every 3 Measures, there will be 16 Divisions”. Therefore the resulting period is a dotted 8th note period.
- $1/3$ = “Every 1 Measure, there will be 3 Divisions”. Therefore the resulting period is half-note triplets.
- $1/12$ = “Every 1 Measure, there will be 12 Divisions”. Therefore the resulting period is 8th-note triplets.
- $1/5$ = “Every 1 Measure, there will be 5 Divisions”. Therefore the resulting period is a quarter-note quintuplet.
- $1/13$ = “Every 1 Measure, there will be 13 Divisions”. Therefore the resulting period is a 13th-note tuplet.

Things can get significantly freakier however:

- $32/25$ = “Every 32 Measures, there will be 25 Divisions”.
- $11/13$ = “Every 11 Measures, there will be 13 Divisions”.
- $512/3$ = “Every 512 Measures, there will be 3 Divisions”.

The order of the numbers in the Sync Panel is chosen based on predicted musical usefulness. Numbers are grouped by their base number, and offer various multiples of two for the given base. (i.e. 3, 6, 12, 24, 48, 96...). The musical result is likely to get more complex and strange the lower you go in the Sync Panel groups.

The beauty of this system is that it offers perfect accuracy both for very short periods as well as very long periods. Additionally, polyrhythms between Input and Output modulation are easy to establish, which will create complex evolving patterns that have composite period lengths that are much longer than their individual components. This is a way to gain incredible organic complexity and evolution from comparatively simple inputs.

There are two different, but similar, techniques to accomplish this:

1. Use a large, equal Measures value for both Input and Output modulation, and then use smaller Divisions values that are different from each other. Ideally these should be co-prime with each other as well as the Measures value to guarantee maximum composite period length. Examples {16/3, 16/5} {64/7, 64/11} {256/13, 256/17}. Using this method will guarantee the entire composite period length is simply the Measures value. This could be used to create shifting textures that perfectly repeat every 16 measures, but have more complexity and variation than is possible to accomplish with a single Image Map and Sync Period. This would also keep the larger composite pattern in phase with the musical structures, which in pop, rock, and electronic music are commonly structured in sections of 4, 8, 16, 32 bars etc.
2. Use different Measures values for both Input and Output. Again, these should ideally be co-prime with each other. The Divisions value can be set as desired. If both Measures values are co-prime, and Divisions values do not create reducible ratios, then the composite period length is the product of the two Measures values. This method is also likely to result in composite period lengths that are co-prime with existing musical structures in the host project, and this will itself create an even more complex composite polyrhythm. Examples:
 - {3/1, 5/1} = Composite period of 15 measures
 - {7/1, 11/1} = Composite period of 77 measures
 - {256/1, 49/1} = Composite period of 12,544 measures!
 - {16/1, 64/1} = Composite period of 64 measures because 16 is not co-prime with 64.

Note: Co-prime is a mathematical term that means two numbers do not share any common factors. For example $9 = 3 \times 3$ and $8 = 2 \times 2 \times 2$, therefore 8 and 9 are co-prime. 12 is not co-prime with either because it has common factors with both: 12

=2*2*3. The general rule, for any numbers, co-prime or not, is the composite period length of the numbers will be the **Least Common Multiple** between them. If this “math stuff” hurts your head, don’t worry; it is not critical to understand. We mention it only for posterity for those who would like to know the nitty-gritty details.

A third, more simple technique regarding using different period lengths for Input and Output modulation is to use a fast/short Sync Period combined with a slow/long Sync Period. The fast pattern will establish some form of thematic phrase or texture, and the long sync pattern can create shifting variations of this phrase or texture. This allows precise control over both micro-level timing events as well as macro-level timing events.

Summary: choosing different Timing Sync Periods for Input and Output creates shifting polyrhythms and textures.

Resource Library Browser Panel

When selecting Tuning Scale, Tuning Waveform, Image Map, or Preset from the Main Page of the GUI Kaleidoscope uses a custom Resource Library Browser Panel to navigate library directories and make the selection. This Browser has been designed to handle libraries with tens of thousands of files or more in the most efficient manner possible.



The Resource Library Browser Panel is a column-style browser that shows both the directory structure as well as the file contents of the final sub-directory folder. Directory selections at each level as well as the final file selection are clearly marked so that it is simple to reconstruct a particular file’s location within the directory at a glance. This makes navigating the incredible vastness of the Resource Library much more manageable. The Resource Library and expansions to the library such as Galbanum Architecture Volume One are well organized by theme and application. Therefore being able to see at a glance where a particular file comes from allows one to make intelligent choices when trying different preset variations by changing the selected file.

The entire directory path is rebuilt within the browser whenever a new preset is loaded. This means whenever a browser is opened, it is immediately evident where a particular file resource has come from, and it makes it a simple task to create “preset mutations” by simply trying other similar resources in the same directory.

The Resource Library Browser Panel has the following features:

- **Directory Browser Columns:** columns on the left of the colored separator line represent sub-folder directories. This area will grow and collapse as needed to display the full path of sub-directories from the root level to the selected file. If more than 32 folders exist in a sub-directory, multiple columns will be used for the sub-directory.
- **File Browser Columns:** columns on the right of the colored separator line represent the file contents of the final selected sub-directory. File selections are made in this area.
- **Show/Hide Folder Button:** toggling the state of this button optionally shows or hides the directory structure. When hidden, only the file contents of the final selected sub-directory folder will be shown. This can be convenient when the sub-directory has a very large number of files in it.
- **Random File:** clicking on this button makes a random file selection from within the sub-directory of the currently active file resource. This is a great way to try different preset variations.
- **Random Folder:** clicking on this button makes a random file selection from within the entire resource library for the current resource type. A random sub-directory will be chosen at each level of the library, and a random file will be selected in the final directory. This is an excellent method to search for “happy accidents”.
- **Shift Left:** shifts all contents of the browser left by one column. This is useful if the final sub-directory contains more files than can be shown within the currently allocated browser area. *Toggling the show/hide folder button will also reset the column shift state.*
- **Shift Right:** shifts all contents of the browser right by one column. This is useful if the final sub-directory contains more files than can be shown within the currently allocated browser area. *Toggling the show/hide folder button will also reset the column shift state.*
- **Lock Panel:** keeps the panel open until manually closed.
- **Close Panel:** manually closes the panel when locked. (When panels are not locked, they will close automatically upon making a file selection as would be common for standard menu selections.)

Mode & View Area

Kaleidoscope’s Mode & View Area contains several buttons that control Resonator Mode, Modulation Modes, and Modulation View as well as offering various display options.

Resonator Mode

Resonator Mode changes the resonator model that is used to process or generate sound in Kaleidoscope, and therefore it has a very large impact on the sound of a given preset.

Kaleidoscope offers the choice of 5 Resonator Modes: two variations of Springs and Strings as well as a FIR mode:

- **1 String:** a String Resonator model as described in detail in the Getting Started section of this manual.
 - The 1 String mode contains harmonics.
 - The 1 String mode produces exponentially decaying delays (in response to an Impulse).
 - The 1 String mode offers frequency dependent Damping.
- **2 String:** a variation of the String Model that offers two differences compared to 1 String:
 - The 2 String mode has an attack envelope which will minimize transients in the output.
 - The 2 String mode has more stop-band rejection for the same Feedback value.
- **1 Spring:** a Spring Resonator model as described in detail in the Getting Started section of this manual.
 - The 1 Spring mode does not contain harmonics.
 - The 1 Spring mode produces exponentially decaying sine waves (in response to an Impulse).
 - The 1 Spring mode offers frequency dependent Damping.

- **2 String:** a variation of the String Model that offers two differences compared to 1 String:
 - The 2 Spring mode has an attack envelope which will minimize transients in the output.
 - The 2 Spring mode has more stop-band rejection for the same Feedback value.
- **FIR:** a pure delay mode. Each line is simply a delayed and gain-scaled copy of the input signal.
 - Technically speaking, this mode is not a resonator.
 - The FIR mode produces delayed and scale impulses (in response to an Impulse).

The filter action of the 2 String and 2 Spring modes is more selective than the 1 String and 1 Spring modes. If for example the White parameter is set to its maximum value to supply the resonators with a pure white noise signal, the “2” modes will suppress the noise signal more and focus the output energy more precisely on the resonant frequencies. This can be highly desirable in many circumstances and is achieved without adding additional decay time. Additionally, these modes create an attack envelope that can also be desirable to minimize noisy transients in musical signals. If it is desirable to perfectly retain transients the “1” modes can be used.

Due to the fact the Strings are more complex spectrally than Springs, it is not typically necessary to build Strings presets that use the maximum number of resonator lines. String-based presets can generally use far fewer lines.

Strings require more CPU and memory resources than Springs. Springs should generally be the go-to choice when designing presets with a huge number of resonator lines.

FIR Resonator Mode

FIR mode effectively creates Finite Impulse Responses. FIRs are simply a summation of Impulses to create a composite impulse response. FIRs have no feedback, and therefore technically no resonance. If all 512 resonator lines are used Kaleidoscope can create a “512 Tap” FIR. The length of the impulse response can be significantly longer however, as the individual tap/impulse locations are set via the Tuning Mode controls, and are therefore typically not spaced at every consecutive sample location. In other words, the (up to) 512 taps can be spread out over time, over much longer ranges: up to 8 seconds in fact. FIR Mode has two general uses:

- **Short FIRs:** FIRs where the total length of the FIR is a couple hundred milliseconds produce complex spatial filter effects which can be use for:
 - Reverb Early Reflection and Ambience Emulation
 - Head Related Transfer Function Emulation
- **Long FIRs:** FIRs where the total length of the FIR is more than a couple hundred milliseconds produce complex spatial delay effects which can be use for:
 - Tempo-Synced delay effects with individual control over each delay tap.
 - Granular-like delay effects

The most interesting aspect of FIR mode is the fact that the Image Maps allow the creation of dynamically changing and evolving FIRs, which is a novel and unusual class of effects processes. That said, FIR mode is fairly experimental, and not the primary use of Kaleidoscope. Predicting exact results especially with short FIRs is quite challenging. Experimentation is the key here.

Modulation Swap

Many of Kaleidoscope’s parameters are independent for Input and Output modulation. Modulation Swap copies all parameters associated with Input Modulation and assigns these values to Output Modulation and vice versa. All Input Modulation settings are now the Output Modulation settings. All Output Modulation settings are now the Input Modulation

settings. This can give a noticeable sonic difference that becomes much more pronounced when Feedback is high for reasons explained in the Getting Started section.

Modulation View Switch

The Modulation View Switch button allows you to select which modulation source is currently displayed and editable: Input or Output. The immediately obvious effect of this control is the image in the Image Map area will change (assuming different images are used for Input and Output), however many other parameter settings are independent for Input and Output as noted elsewhere in the manual. Many of these parameters use a shared control. These shared controls (including knobs, buttons, and even option lists inside of panels) will display and allow you to edit the value of the parameter associated with the actively displayed Modulation View. For example, if you are viewing Input, and you change the Mod Strength knob, you are changing Mod Strength for Input, not Output. Switching the view to Output would allow you to edit the Output Mod Strength and the knob will automatically update itself to the correct value when changing views. This is quite intuitive in use.

Modulation Modes

Kaleidoscope has two Modulation Mode buttons: one for Input Modulation, and one for Output Modulation. These two buttons can independently have one of three states:

- **Dynamic:** the Image Map is scanned left to right over time as explained thoroughly in the Getting Started section.
- **Static:** the value in the left-most pixel column in the current Image Map is held statically over time. There is no modulation over time, but the static values can be used to control the gain and spatial position of each resonator.
 - X-Offset can be used to change which pixel column is the “left-most” and therefore simply dragging X-Offset can give 1024 different variations of static modulation from a single Image Map
 - Y-Offset can be used to sift pixel rows vertically giving 512 possible variations.
 - Combining X-Offset and Y-Offset gives over half a million different possible static modulation maps from a single image!
 - The Image Map is drawn in a special manner in this mode that repeats the left column over the width of the image map to make it more readable. This display updates when changing offset values.
 - **Static Mod Mode requires significantly less CPU resources than Dynamic Mod Mode.**
 - **Static mode can be thought of as a 512-band stereo graphic EQ.**
- **Off:** Modulation is disabled completely. This eliminates the CPU usage requirements of modulation for either Input or Output Modulation and allows the result to be solely controlled by the other active Modulation source.
 - A special disabled state image is shown in the Image Map area to communicate its disabled state.
 - If Both Input and Output are set to Off the Wet output of the plug-in is muted.

Line Count Displays

Below the Modulation Mode buttons are three numbers. These numbers communicate the Line Counts:

- **Defined Lines:** the left value communicates how many lines are defined by the current tuning.
 - $\text{Defined Lines} = (\text{Number of ratios in Scale or waveform}) * \text{Partials} * \text{Duplicates}$
 - Defined Lines is limited to the Image Map Height: 512
 - Defined Lines is meant to communicate information about the size of the tuning used
- **Enabled Lines:** the center value communicates how large the **potential** canvas size is.
 - Enabled Lines is the smaller of the value of Defined Lines and the Max Enabled Lines Slider
 - The Enabled Lines area is the part of the Image Map that is not covered by the semi-transparent “grayed out” Disabled Area

- The Enabled Lines area represents the number of resonators that are potentially available to use
- In most cases, the Enabled Lines area will match the Max Enabled Lines Slider; however for Scale Tunings that use very few ratios the Enabled Lines area may be smaller than the value of the slider.
- **Active Lines:** the right value communicates the number of Lines that are actually in use and consuming CPU power
 - $\text{Active Lines} = \text{Enabled Lines} - \text{Empty Lines} - \text{Frequency Out Of Bounds Lines}$
 - **Empty Lines** are lines in the Image Maps where all pixels for the current row are black or off. Since these lines are completely silent they are made inactive and do not consume CPU resources.
 - **Frequency Out Of Bounds Lines** are lines that have frequencies greater than the maximum or less than the minimum allowable resonator frequency. These lines are automatically made inactive and do not consume CPU resources. These lines are also drawn with the special semi-transparent overlay to communicate that they are outside the allowable frequency range. There is no problem with having such lines in a preset; they simply are not used and therefore we make them inactive to conserve CPU resources.
 - **Active Lines is a direct predictor of the amount of CPU resources used by the current preset.** (Other factors also affect CPU usage, but the Active Lines number is more or less a linear factor effecting CPU usage. 100 Active Lines will use 100 times more CPU resources than 1 Active Line if all other settings are the same when comparing these two states.)

Soft & Mod Curve Displays

The Soft & Mod Display shows the effective transfer function of the Mod controls as well as the Soft controls.

- **Mod:** the Mod line curve represents the transfer function that remaps pixel brightness and thus modulation envelope amplitude values depending on the values of Mod Curve, Mod Strength, and Mod Offset. A 45-degree straight line represents the nominal, no-change state, and in this state pixel data is used exactly as it is found on disk in the active Image Map. Other settings of these controls manipulate and remap this data in useful ways. The Mod display shows the details of how this remapping is currently applied. The horizontal axis represents the input value, and the vertical axis represents the output value. Thus the display is similar to transfer function displays found in audio signal waveshapers.
- **Soft:** the Soft line curve represents the represents the transfer function that adjusts resonator line gain as a function of line frequency and the values of Soft, Soft Function, and Soft Power. This is generally used to reduce the gain of high frequencies. The Soft line curve shows the relative decrease in line gain. Horizontally represents the resonator line in the Image Map, vertically represents the relative gain adjustment to this line. A straight horizontal line across the top of the display represents the nominal, no change state.

Image Map Area Display Options

The Image Map Area Display Options control what data is shown in the Image Map Area.

- **Image Map Display Option:** shows or hides the Image Map.
- **Tuning Function Display Option:** shows or hides the Tuning Function.
- **X-Grid Display Option:** shows or hides the X-Grid.
- **X-Grid Value:** allows the size of the X-Grid to be entered in Pixels
- **Y-Grid Display Option:** shows or hides the Y-Grid.
- **Y-Grid Value:** allows the size of the Y-Grid to be entered in Pixels.

X & Y Grids

The X and Y Grids are useful visual references to show sub-divisions of time and frequency space. They can be set to intelligently match the organization of a given Image Map and Tuning. For example if an Image Map is of a rhythmic nature and has 16 “pulses” in it, the X-Grid could be set to 64 pixels (1024/16) to match the existing rhythmic base. The Y-Grid could be used to mark octaves, or perhaps notes if a significantly number of Partials and Duplicates are used with a musical Tuning. For example if Partials=8 and Duplicates=8, the Y-Grid might be set to 64.

X-Offset, Y-Offset, and X-Size can be “snapped-to-grid” by holding down the Shift key on the keyboard while dragging.

X-Grid and Y-Grid values are saved into and recalled from presets.

Meter & Mix Area

Kaleidoscope’s Meter & Mix Area contains controls and displays relating to gain levels within the plug-in.

Dry Gain

Dry Gain sets the level of the dry input signal in Kaleidoscope’s output.

Wet Gain

Wet Gain sets the level of the wet signal, i.e. the summation of the output of all resonators, in Kaleidoscope’s output. We offer separate Wet and Dry gains instead of a Mix control in Kaleidoscope due to the fact that the energy level in Kaleidoscope’s wet signal can be rather unpredictable due to the nature of extreme resonance, particularly when the input is highly tonal and contains narrow spectral peaks that align with the tuning of the resonators, i.e. a sustained violin note that matches a resonator line tuning in Kaleidoscope. Therefore, since the average energy of the Wet signal cannot always be guaranteed to be equal with the Dry signal, separate control over both is a better solution.

White

Kaleidoscope’s White control is a very important and useful special feature. It effectively mixes white noise into the input of the resonators in a specialized manner. The utility of this feature is twofold:

- *White can be used to subdue extreme changes in gain in resonator output in cases when resonator tuning aligns with highly tonal input signals with narrow spectral peaks.*
- *White can be used at its maximum setting to transform Kaleidoscope into a pure synthesis device or instrument!*

The White control works in the following manner:

- 0%: resonator input is the incoming Dry signal as expected.
- $0 < \text{White} \leq 100\%$: the dry signal is amplitude modulated by white noise.
 - At less than 100% this has the effect of significantly broadening the spectrum of the input signal, and guaranteeing all of Kaleidoscope’s active resonators will be excited significantly.
 - At exactly 100% the spectrum of the input to the resonators is exactly flat. This has the effect of producing enveloped white noise.
 - *At 100% all pitch information in the input signal is “erased”, but all rhythmic information in the input is perfectly retained. Kaleidoscope’s resonators will supply a new tonality based on current*

Tuning settings, but rhythmic information will be an interaction between the input and Kaleidoscope's Image Maps

- 100 < White <= 200%: pure white noise is mixed with the amplitude-modulated signal.
 - At less than 200% this begins to weaken the influence of the rhythmic pulse from the input signal
 - At 200% the input signal is completely gone, and only the internal white noise generator remains
 - ***At 200% Kaleidoscope is effectively transformed into a self-contained synthesizer and sound generator and no longer requires any input signal! This can be a very powerful way to work with Kaleidoscope.***

Mix Lock

The Mix Lock control locks the values of Dry, White, and Wet, so that their values remain fixed when changing presets. This is useful for example if you would like to generate new content by using the White feature at 200% for example, or know that you do not want any Dry signal in output.

Image Map Area

Kaleidoscope's Image Map Area holds the Image Map as well as several sliders and buttons related to it.

Image Map

The Image Map is a display of the currently viewable Image Map. This area displays the Image Map for either Input or Output as controlled by the Modulation View Switch. This area also shows the Tuning Function, X-Grid, Y-Grid, special disabled states for the Image Maps when Mod Mode is Off, as well as special Tuning Function background when the Image Map view is hidden. All of these features have already been discussed in detail earlier in this manual. See the previous descriptions if more information is needed.

Max Enabled Lines Slider

The Max Enabled Lines Slider provides manual control over the number of Enabled Lines in a given preset. We call it Max Enabled Lines instead of simply Enabled Lines because in certain cases of Tuning settings, the Enabled Lines area will be less than the slider; namely Enabled Lines can not be greater than Defined Lines, so in cases where Defined Lines is small (i.e. using a 3-note Tuning Scale without any Partial, Duplicates, Octaves etc.) the Enabled Lines value may be less than the slider. In most cases Enabled Lines is equal to Max Enabled Lines.

The Max Enabled Lines Slider gives you direct control over the amount of CPU resources used by the current preset. To reduce CPU usage simply move the slider lower, thus reducing the number of Enabled Lines.

Y-Offset Slider

The Y-Offset Slider allows you to shift the vertical position of an Image Map. This is done a cyclic fashion that wraps the lines at the top of image that are pushed outside of the Image Map area back down to the bottom of the image, thus always preserving a fully populated Image Map area. The majority of the Image Maps in the Resource Library have been designed to tile perfectly so that any arbitrary Y-Offset values are free of discontinuous "seams" assuming the nature of the image is meant to be continuous.

- Holding down the shift key on the keyboard while the slider will snap offset amounts to the Y-Grid value.
- The Image Map itself can also be freely dragged as well as snapped-to-grid to change Y-Offset.
- Y-Offset can achieve 512 variations of a single Image Map.

X-Offset Slider

The X-Offset Slider allows you to shift the horizontal position of an Image Map. This is done in a cyclic fashion that wraps the lines at the right of image that are pushed outside of the Image Map area back to the left edge of the image, thus always preserving a fully populated Image Map area. The majority of the Image Maps in the Resource Library have been designed to tile perfectly so that any arbitrary X-Offset values are free of discontinuous “seams” assuming the nature of the image is meant to be continuous.

- Holding down the shift key on the keyboard while the slider will snap offset amounts to the X-Grid value.
- The Image Map itself can also be freely dragged as well as snapped-to-grid to change X-Offset.
- X-Offset can achieve 1024 variations of a single Image Map. Combined with Y-Offset this gives over half-a-million variations from a single Image Map!

X-Size Circle

The X-Size Circle is used to change the effective size of the Image Map. Clicking and dragging the circle adjusts X-size.

- Double Clicking on the circle will change its appearance to have double-rings instead of a single-ring. In this mode, the circle can be dragged vertically as well to adjust both X-Size and Enabled Lines.
- Holding down the shift key on the keyboard while dragging the circle will snap its position to the current X and Y grid values

X-Size values smaller than the full width of the Image Map can be useful to crop an Image Map to use only part of the image. X-Size can also be used to establish Polyrhythms as discussed in the description of the Scanning Mode control earlier in the manual.

Image Resize Y

Image Resize Y “contracts” the height of the Image Map by various power-of-two factors. At its nominal, default value the Image Map is not contracted and uses the full height of the Image Map area. Additional states contract the image to be 256, 128, 64, etc. lines tall. This is accomplished simply by using every other line of the image at each reduction step. The top of the image is then filled with empty black lines. Image Resize Y is useful when using smaller numbers of Enabled Lines. For example if you have used the Max Enabled Lines slider to reduce the Enabled Lines area to a height of 128 lines, you might like to use Resize Y as well so that the entire “general feel” of the Image Map remains the same.

Image Orientation

Image Orientation performs Image Map transformations including flipping the horizontal axis, flipping the vertical axis, and rotating axis. There are 8 possible states as explained visually with the following image:

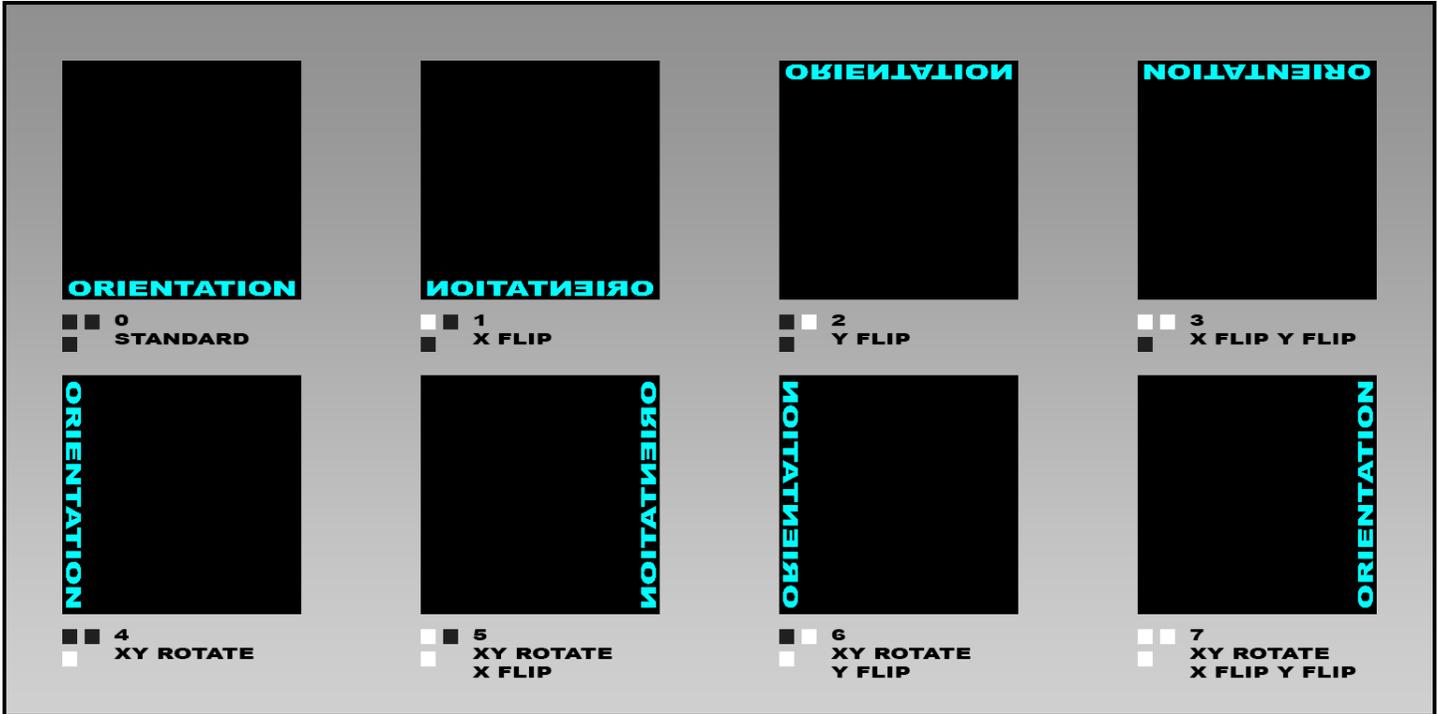


Image Offset combined with X-Offset, Y-Offset and Image Inversion provide over 8 million ways to use a single Image Map! Manipulations to Image Resize Y, X-Size, and Max Enabled Lines move this count well into the astronomically huge numbers! All from a single Image Map!!

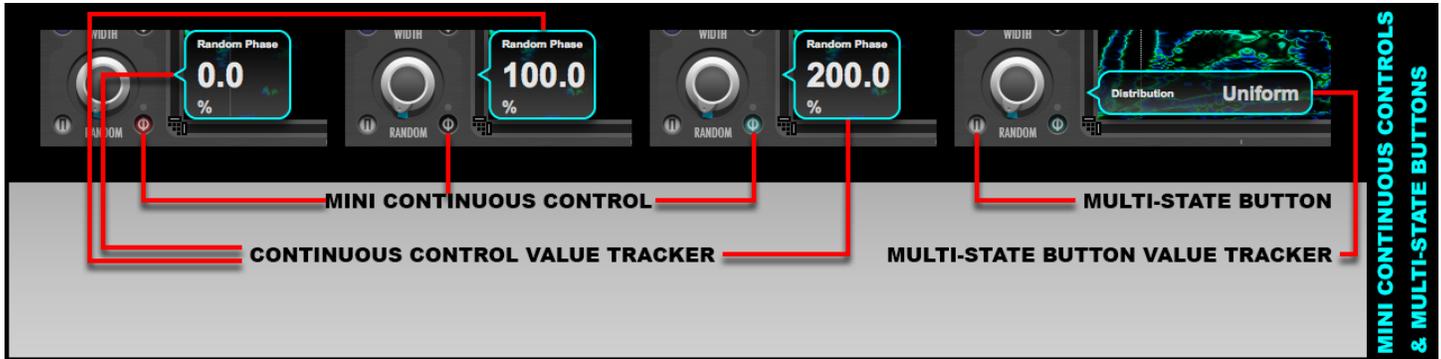
The Image Orientation state is persistent across making other changes to the Image Map such as loading a new Image Map Resource File. Therefore, the Resource Library can be easily browsed while all selections are automatically subjected to the same transformation.

Main Page Controls Areas

On both sides of the Image Map area are two symmetrical sets of controls each consisting of four large knobs, and 9 smaller supporting controls. The controls on the right of the Image Map handle various frequency-related filtering behavior. The controls on the left handle everything else. The controls are grouped into a primary control in the form of a large knob, and two or three supporting controls in smaller multi-state buttons, or miniature continuous controls.

The miniature continuous controls have a small dot above them on the GUI to differentiate them from the multi-state buttons. These controls behave like knobs: clicking and dragging on them can change their values and double-clicking on them allows text entry. They change in color hue and brightness as their value changes to roughly communicate their current values at a glance.

Additionally, all controls found on the Main Page of the GUI have optional “Pop-Up Value Trackers” which display the parameter name, exact numerical value, and units when applicable. Pop-Up Value Trackers for Knobs and miniature continuous controls look different than those for multi-state buttons, and this is another simple way to differentiate these type of controls.



Feedback

The Feedback control adjusts the overall base level of feedback for all resonators. As thoroughly explained in the Getting Started section of the manual:

- **High Feedback** creates:
 - More resonance, higher frequency selectivity
 - Longer decay times
 - Approximately pure sine waves at maximum values when using Springs
 - Approximately pure harmonics at maximum values when using Strings
- **Low Feedback** creates:
 - Less resonance, lower frequency selectivity
 - Shorter decay times
 - Band-pass filter-like behavior

A Feedback value of 50% creates a base RT60 decay time of approximately 1.0 second.

Relative Feedback

The main Feedback knob establishes a base Feedback amount. This is the value that is used at Tuning Ratios of 1.0, which are typically (but not always) the lowest ratio in Tuning Scale files. Relative Feedback allows the Feedback amount of each resonator to be scaled inversely by its Tuning Ratio by a variable degree.

Relative Feedback allows high frequencies to have shorter decay times than low frequencies, as commonly occurs in nature.

- **0%:** Relative Feedback results in decay times that are scaled by the reciprocal of the given Tuning Ratio. If a 1.00 Tuning Ratio produces a 1.0 second decay time, a 2.00 Tuning Ratio would produce a 0.50 second decay time.
- **100%:** All resonators have exactly the same decay time which is the full length value established by the main Feedback Knob.
- **Low Values:** tend to sound more natural as this is the natural tendency for most real-world physical objects
- **Higher Values:** produce more synthetic results where high frequencies tend to sustain longer than one would commonly expect in natural materials. This can be useful however to achieve more precise spectrums and Output Modulation can be used to artificially stop the resultantly long decay times.

Feedback Mode

In addition to scaling a resonator's Feedback by its *relative* Tuning Ratio, it may also be scaled by its *absolute* Hertz Tuning Reference value.

- **Equal:** Feedback is equal for any/every Tuning Reference

- **Relative:** Feedback is scaled by the Tuning Reference. High frequencies have less feedback than low frequencies.

If for example Relative Feedback is at 0%, and the Reference Frequency is A0, harmonics in this tuning will decay faster than the fundamental frequency. However, if the Reference Frequency is changed to A6, and Feedback Mode is Equal, the overall decay times for the fundamental and the harmonics will be same for both A0 and A6. If Feedback Mode is set to Relative, A6 will have substantially shorter decay times.

This Relative behavior is also common in nature. The highest Strings on a piano or guitar tend to have shorter decay times than the lowest Strings for example. Using the Relative value is helpful when rendering multiple “takes” of the same preset at different pitches for example.

The Feedback Mode and Relative Feedback are independent of one another. Their effects are similar but not identical. Relative Feedback is affected by each resonator’s relative Tuning Ratios. Feedback mode is affected by changes to absolute Hertz values of the Tuning Reference, regardless of a particular resonator’s Tuning Ratio.

Feedback Sign

When using the String resonator modes, Feedback Sign affects harmonic structure of the resulting resonator spectrum.

- **Positive:** produces all harmonics as explained thoroughly in the Getting Started section of this manual.
- **Negative:** shifts the fundamental frequency down one octave and produces only odd harmonics.

Negative can be used to produce triangle and square-wave like timbres as well as roughly emulate woodwind instruments with the proper use of the Damp filter.

When using the Spring resonator modes, Feedback Sign affects the initial phase of the resulting sinusoid component in the impulse response and therefore switches the filter between Low Pass and Band Pass.

- **Positive:** results in Sine phase, which technically gives a Low Pass filter response.
- **Negative:** results in Cosine phase, which technically gives a Band Pass filter response.

When Feedback is high, or when using the 2 Spring resonator mode, the sonic difference between these two is quite small; however, when feedback is low and the resonator mode is set to 1 Spring, there is a clearly audible difference. The Negative, Band-Pass response will preserve more transients in the signal and its band-pass nature, compared to the low-pass nature of the positive value, will be more apparent.

Feedback Sign has no meaning for the FIR resonator mode and is therefore disabled.

Mod Strength

Pixel brightness of each pixel in each line of an Image Map is mapped to an amplitude value on a modulation envelope as explained thoroughly in the Getting Started section of this manual. Pixel brightness levels use a linear scale in image files saved on disk. Mod Strength can manipulate and remap this data onto non-linear scales in a way similar to the way waveshaping works on audio signals. This can be used to effectively do things such as adjust the contrast in an image by bringing down low levels significantly while maintaining or even boosting the higher levels. In the context of Kaleidoscope use of Image Maps this kind of manipulation allows adjustments to the dynamics of the resulting modulation “performance”. It can be used to increase or decrease existing accents in an Image Map, for example. It can be used to effectively thin dense images, by reducing the brightness, and thereby gain, of all but the brightest areas of an Image. Mod Strength is a bi-polar control. Generally, negative values result in less overall brightness, and positive values result in more overall brightness, but this is partially dependent on the selected Mod Curve.

Mod Curve

Mod Curve is a multi-state button that selects the transfer function that is used by the Mod Strength knob to remap pixel brightness to nonlinear curves.

- **Power:** a variable power curve.
- **Exponential:** a variable exponential curve.
- **CosPower:** a variable Cosine Power curve, which gives behavior like Power but with an “S” shaped curve.
- **ArcTan:** a variable Arc Tangent and Tangent curve. Negative values gives Arc Tangent, Positive Values give Tangent.
- **Root:** a variable root curve.
- **Bell:** a variable asymmetrically raised cosine envelope that can target specific ranges of brightness to retain.
- **Stripes:** produces stripes or banding in smooth gradients. Useful for create effects and to add complexity.

The best way to get a feeling for these various curves is to cycle through them and move the Mod Strength knob while looking at the Soft Display and the resulting changes to the Image Map.

Mod Offset

Mod Offset can scale the vertical range of the Mod transfer function, effectively limiting the minimum brightness value a pixel can have. If Mod Offset is 50% from example, the pixel brightness range will become 50% to 100%.

Note: Mod Offset values of anything greater than exactly 0.0% will make all lines “non-empty”. If some lines were empty previously, containing only exactly black 0.0% pixels, these pixels will no longer be exactly black. Thus these lines will no longer be empty, and thus they may potentially become Active. Therefore mod Offset can potentially increase CPU usage.

Image Width

Color is used in Kaleidoscope to control gain-based stereo panning as thoroughly described in the Getting Started section of this manual. The ability to have individual voices (resonator lines) independently moving around in spatial position is a wonderfully powerful feature; however, sometimes too much of a good thing is a bad thing. The Image Width control addresses this situation.

Image Width effectively scales the maximum difference between the modulation envelope gain levels of the left and right channel. This has the effect of collapsing width created by differences in left and right Image Map channels. Visually areas of distinct, fully separated primary-hue color in the image will converge, and shift to the secondary color created by the equal mix of the two. (i.e. depending on the selected Color Scheme separated red and green areas will become yellow, separated green and blue areas will become cyan, separated blue and red areas will become magenta). If there are no hue differences in the Image Map, this means there are no stereo differences in the modulation envelopes.

- **-100%:** Fully separated color channels, and therefore maximum width. Left and Right channels are swapped.
- **0%:** Color channels are mixed together, and therefore there is no width.
- **100%:** Fully separated color channels, and therefore maximum width. Left = Left, Right = Right.

Tip: if using Kaleidoscope in a music project that is comprised of many component tracks, consider using different width values for each. Every single track in a project does not need to have maximum width. Even 50% width is generally sufficient to give the impression of spatial motion.

Input Cross

Kaleidoscope is a stereo-in, stereo-out process. If the input signal is already highly spatialized this will be reflected in Kaleidoscope’s output. If Input Cross is 0%, which is its default value, Kaleidoscope is effectively a dual-mono process. The left input channel will be processed only by the Left bank of resonators, and will be effected only by the Left Image

maps. The same is true for the right channel. Input Cross can cross-mix the input channels to feed some of the left channel into the right channel and vice versa.

- **0%**: Fully separated input channels. Left = Left, Right = Right.
- **100%**: Input channels are mixed together, and therefore the input becomes mono and spatial information in the input is disregarded.
- **200%**: Fully separated input channels. Left and Right channels are swapped. Left = Right, Right = Left.

Input Cross effectively controls how existing spatial information in the stereo input signal is reflected in Kaleidoscope's wet output.

Phase Width

Kaleidoscope offers a third way to manipulate stereo width of its output: Phase Width. Phase Width controls the phase correlation between Kaleidoscope's Left and Right channels.

- **0%**: No phase differences between the Left and Right channels.
- **100%**: Statistically equal parts in-phase and out-of-phase, producing zero phase correlation.
- **200%**: Exact out-of-phase/anti-phase/180-degree phase differences between the Left and Right channels.

Statistically speaking, 100% would be the expected result if two stereo microphones were used in a natural recording, and this produces the greatest natural width. Phase Width of greater than 100% can be used for special effects that jump out of the speakers or seem to come from improbable spatial locations such as behind your head. Phase Width of less than 100% can be used to focus the energy more towards the center of the stereo sound field. This can be beneficial for bass-heavy sounds where mono-compatibility might be a concern. Additionally, as with Image Width, in complex arrangements, it is often beneficial to use different Phase Widths for different track components.

Tip: In order to stickily guarantee that Kaleidoscope's output is exactly mono, Image Width must be 0%, Input Cross must be 100%, and Phase Width must be 0%. All three controls affect the overall phenomenon of Width perception.

Random Pitch

The Random Pitch knob can randomly detune the frequency of all active resonators by varying amounts. This can be used to create subtle chorus-like effects, particularly when used in conjunction with Tuning Duplicates.

- **0.0%**: No detuning.
- **>0.0-25%**: Very subtle detuning suitable to emulate micro-pitch-shift-doubling effects
- **25-50%**: Subtle chorus-like effects
- **50-75%**: Less subtle chorus effects
- **>75%**: strong detuning and pitch randomization will start to confuse the identity of most musical scales

Random Distribution

The Random Distribution button switches between two different random distributions for random values:

- **Gaussian**: The probability density function follows a standard Gaussian Bell curve. Random values are statistically more likely to be found closer to the central value than at the extremes. This is the way most natural systems operate in the real world.
- **Uniform**: The probability density function is equal for the entire range. Random values are equally likely to be chosen from anywhere within the range. This tends to generate a slightly stronger effect for the same random range as compared to Gaussian.

Random Phase

Random Phase randomizes the phase of resonators by varying amounts. When summing a huge number of resonators or oscillators or general audio signals, if they all have the same phase this can result in large peak values in the sum. Randomizing the phase of each component can minimize this to a large degree. Additionally, we use this control to hide an Easter Egg to find out who really takes time to read this manual: at values of over 100% this control adds random delays to each resonator line which get quite extreme at the maximum 200% value and be used to create granular-like effects. This can also given an effect similar to Early Reflections in reverb.

- **0.0%:** No Phase randomization.
- **100%:** Optimal phase randomization.
- **>100%:** Moving into extreme random-delay territory to achieve granular-like effects.

Damp Frequency

The Damp Frequency Knob controls the frequency of the damping filter used in Kaleidoscope's String resonator models. It has no effect in the Spring modes, and is disabled. Higher values translate to a higher filter cutoff frequency. The exact frequency used by the Damp filter for each resonator is dependent on the Damp Mode.

Damp Mode

The Damp Mode button switches between two different methods used to set the exact frequency of the damping filter:

- **Fixed:** the damp frequency is fixed at an equal value in Hertz for all resonator lines regardless of their tuning. This is useful when using the Strings mode to model large spaces and create reverb-like presets.
- **Relative:** the damp frequency is set relative to the particular resonator's tuning. This is effectively like key-tracking in a synthesizer. This will preserve the same relative timbre and harmonic structure for each and every resonator regardless of its tuning and can be quite useful.

Tip: Relative Damp mode with low Damp Frequency values can be a great choice when using a large number of active String resonators. This will significantly reduce the energy in high harmonics and allow only the first few low-order harmonics to be prominent, thus keeping more available free space in the spectrum that can be occupied by other resonators.

Damp Type

Kaleidoscope offers three different filter choices for Damping:

- **Off:** No damping filter is used.
- **6dB LP:** a first order low pass filter that offers a -6dB per octave slope
- **12dB LP:** a second order low pass filter with variable Q that offers a -12dB per octave slope
- **Band Pass:** a band pass filter with variable bandwidth

Tip: by using the Relative Damp mode with the Band Pass Damp Type, it is possible to target specific harmonics of the String resonator. In this scenario, the Damp Frequency value can be thought of as the harmonic number: a value of 1.0 is the fundamental frequency; a value of 3.0 is the third harmonic; and so on. This can produce quite a wide range of potential timbres.

Extra Credit Nerd Tip: If Feedback sign is Negative and you would like to target specific harmonics using Relative Damp mode with the Band Pass Damp Filter you should actually use (0.5, 1.5, 2.5, 3.5...) because Negative feedback shifts pitch down one octave and produces only odd harmonics.

Damp Q

The Damp Q control adjusts the Q or Bandwidth of the Damp filter when using the 12dB LP or Band Pass filter options.

For the 12dB LP Mode:

- **<0.0%:** more gentle filter slopes
- **0.0%:** the maximally flat, Butterworth filter condition
- **>0.0%:** steeper filter slopes, and a resonant peak around the cutoff frequency

For the Band Pass:

- **<0.0%:** wider bandwidth
- **0.0%:** nominal one-octave bandwidth
- **>0.0%:** narrower bandwidth

Note: some settings of Damp Frequency, Mode, Type, and Q can cause some minor degree of detuning to the String and pitch perception may not be 100% exactly as expected. The reason for this is that the damping filter itself creates a dispersion effect that shifts the exact tuning of some of the harmonics. The fundamental frequency may be perfectly tuned, but harmonics may be slightly out of tune and therefore pitch perception, which is a complex psychoacoustic process, may be affected. This exact same phenomenon occurs in real String instruments as well however, so generally this is not a problem. "Stretch Tuning" used to tune grand pianos was developed to address this phenomenon for example. If anything, the presence of this phenomenon in Kaleidoscope augments its connection to the laws of the physical universe, and thus we consider this more of a feature than a design flaw. Oversampling reduces this to some extent.

Soft

By default each resonator in Kaleidoscope has equal gain before the effects of the Image Map. In many natural systems however, high frequencies have less energy than low frequencies. In fact much of the natural world follows something close to a "1/F rule": gain is scaled by the reciprocal of frequency. This is effectively a pink-noise spectrum and is very common in the natural world. Indeed the human auditory system perceives pink spectrums as being roughly flat perceptually. Therefore in Kaleidoscope, it can be highly beneficial to have an easy way to scale the gain of resonators based on their respective frequency. This is exactly what the Soft controls do: Soft scales the gain of each resonator line based on its relative Tuning Ratio, and thus its frequency.

- **Low Values:** approach equal gain for all resonators, retaining equal energy at all resonator frequencies
- **High Values:** reduce the gain of high frequencies to give a less bright result

Note: the best position of the Soft control is highly dependent upon the input signal. Kaleidoscope is like a selective mirror or prism as previously explained. If it is fed white noise from the internal generator, and a wide range of resonator frequencies are used, the resulting overall frequency contour of the output will tend towards white and in such cases judicious use of the Soft control may be helpful to achieve a less bright result. If however the input signal is already deficient in high frequency content, it may be completely unnecessary to lower it even further via the Soft control. Soft is similar to an EQ; your ears are your best guide.

Soft Function

The Soft Function multi-state button offers several options to choose from that affect exactly how resonator frequency is translated into a respective line gain.

- **Harmonic:** an inverse harmonic power series.
 - At Soft = 100%, this mode produces exactly a 1/F frequency weighting.

- **Exponential:** an exponential curve which tends to be less severe at low frequencies and much more severe at high frequencies compared to Harmonic mode.
- **Linear:** a linear curve which tends to be less severe at low frequencies and much more severe at high frequencies, potentially muting some frequencies completely, compared to Exponential mode.
- **Butterworth:** a curve emulating the frequency response of a classic maximally flat, Butterworth low pass filter: approximately flat up to the cutoff point, and then sharply falling
- **Asym Bell:** an asymmetrical bell curve that actually decreases gain at very low frequencies around the fundamental, smoothly rises to a peak, and then smoothly falls back down as frequency gets significantly higher.

The value Soft knob affects the strength of these various functions. The resulting Soft transfer function is shown in the Soft Display.

Soft Power

The Soft Power multi-state button offers two additional refinements to the behavior of the Soft controls. Each state offers a variation of the current Soft Function.

- **I:** variation one is generally less severe
- **II:** variation two is generally more severe

Hi Cut Frequency

The Hi Cut controls offer control over a simple high cut (Low Pass) filter that is placed at Kaleidoscope's input and/or output. This gives another method to tame excessive high frequencies in Kaleidoscope's wet output. The Hi Cut filter is a standard stereo filter and works on absolute frequency basis like common filters. This differs from the Soft control that works on a relative basis. Hi Cut Frequency is set in Hertz.

- **Minimum:** 8 Hz
- **Default:** 4096 Hz
- **Maximum:** 32,768 Hz (Limited to $\frac{1}{2}$ of the current sample rate)

Hi Cut Type

The Hi Cut filter offers several filter types:

- **Off:** The filter is disabled
- **6dB:** a first order filter with a 6dB per octave slope
- **12dB:** a second order, maximally flat, Butterworth filter with a 12dB per octave slope
- **24dB:** a fourth order, maximally flat, Butterworth filter with a 24dB per octave slope

Hi Cut Routing

The Hi Cut Routing multi-state button offers three different filter routing choices:

- **Input:** The filter is inserted into Kaleidoscope's input before the resonators.
- **Output:** The filter is inserted into Kaleidoscope's output before the resonators.
- **In & Out:** The filter is used at both input and output locations effectively doubling its order and slope.

Input and Output locations effectively produce the same result in most cases. The most noticeable difference occurs when using presets with extremely fast modulation rates that intentionally cause aliasing for distortion effects.

In & Out doubles filter order and slope, and changes the filter response curve from Butterworth to what is known as Linkwitz-Riley. The filter response is down -6dB at the cutoff frequency in this mode instead of -3dB like the other two modes.

Lo Cut Frequency

The Lo Cut controls offer control over a simple low cut (High Pass) filter that is placed at Kaleidoscope's input and/or output. This gives another method to tame excessive low frequencies in Kaleidoscope's wet output. The Hi Cut filter is a standard stereo filter and works on absolute frequency basis like common filters. This differs from the Soft control that works on a relative basis. Lo Cut Frequency is set in Hertz.

- **Minimum:** 8 Hz
- **Default:** 64 Hz
- **Maximum:** 32,768 Hz (Limited to ½ of the current sample rate)

Lo Cut Type

The Lo Cut filter offers several filter types:

- **Off:** The filter is disabled
- **6dB:** a first order filter with a 6dB per octave slope
- **12dB:** a second order, maximally flat, Butterworth filter with a 12dB per octave slope
- **24dB:** a fourth order, maximally flat, Butterworth filter with a 24dB per octave slope

Lo Cut Routing

The Lo Cut Routing multi-state button offers three different filter routing choices:

- **Input:** The filter is inserted into Kaleidoscope's input before the resonators.
- **Output:** The filter is inserted into Kaleidoscope's output before the resonators.
- **In & Out:** The filter is used at both input and output locations effectively doubling its order and slope.

Input and Output locations effectively produce the same result in most cases. The most noticeable difference occurs when using presets with extremely fast modulation rates that intentionally cause aliasing for distortion effects.

In & Out doubles filter order and slope, and changes the filter response curve from Butterworth to what is known as Linkwitz-Riley. The filter response is down -6dB at the cutoff frequency in this mode instead of -3dB like the other two modes.

Browser Page Parameters & GUI Details

The Preset Browser

The preset browser page displays up to 224 world-class, professionally designed factory presets that are only a single click away. A quick scan through these presets will allow you to quickly find exactly the right sound you need for your application.

Kaleidoscope 1.0 ships with approximately 1,000 factory presets. More presets may become available in future as well. As such, Kaleidoscope offers a huge number of factory presets to choose from, and management and organization of these presets becomes important to maintain maximum ease of use. In order to achieve that goal Kaleidoscope factory presets are organized into the following folders:

- **1.1 Rhythmic Melodic**
- **1.2 Rhythmic Chordal**
- **1.3 Rhythmic Harmonic**
- **1.4 Rhythmic Atonal**
- **1.5 Rhythmic Filter**
- **2.1 Nano Groove Melodic**
- **2.2 Nano Groove Chordal**
- **2.3 Nano Groove Harmonic**
- **2.4 Nano Groove Atonal**
- **2.5 Nano Groove Filter**
- **3.1 Textural Melodic**
- **3.2 Textural Chordal**
- **3.3 Textural Harmonic**
- **3.4 Textural Atonal**
- **3.5 Textural Filter**
- **4.1 Ambient Melodic**
- **4.2 Ambient Chordal**
- **4.3 Ambient Harmonic**
- **4.4 Ambient Atonal**
- **4.5 Ambient Filter**
- **5.1 Static Melodic**
- **5.2 Static Chordal**
- **5.3 Static Harmonic**
- **5.4 Static Atonal**
- **5.5 Static Filter**
- **6.1 FX Volume Envelope:** trance gating, ducking, pumping, and general single line volume automation effects
- **6.2 FX Delay:** tempo synched delay effects
- **6.3 FX Granular:** delay effects with effectively random, non-musical delay times
- **6.4 FX AM Distortion:** distortion effects created by extreme amplitude modulation and intentional aliasing created by extreme modulation rates

The general idea of the folder list is to make it easier to find what you are looking for. Excluding the four special effects groups described above, organization is based on a pairing of two classifications: one for organization in time, and one for organization in frequency. Time classifications are as follows:

- **Rhythmic:** characterized by the strong presence of traditionally musically meaningful discrete pulses in time
- **Nano Groove:** characterized by highly complex rhythmic material with very fast textural phrases and polyrhythms
- **Textural:** characterized by the presence of well defined time structures that do not follow standard musical rules
- **Ambient:** slow and evolving soundscapes and drones, typically with high feedback and no obvious transients
- **Static:** characterized by the complete lack of changes over time

Frequency classifications are as follows:

- **Melodic:** tonality is based on musical scales or modes, including non-western varieties
- **Chordal:** tonality is based on musical chords as dictated by the western classical music and jazz traditions
- **Harmonic:** tonality is based on the harmonic series or some sub-set therefore and should be considered musically as a the spectrum of a single note
- **Atonal:** tonality is based on non-musical rules such as arbitrary mathematical functions, waveforms, spectrums, and anything else that is not easily recognized as traditionally musical frequency organization
- **Filter:** characterized by the complete lack of strong tonality and resonance due to very low feedback

Package & Sub-Folder List

The Preset Browser Package and Folder List is found in the left column of the Preset Browser. This area has two different levels of directory organization: one called Packages, and another called Sub-Folders. Packages are a collection of a single level of Sub-Folders, which comprise a particular collection of presets such as the Factory presets, the User presets, and Preset Expansions as they become available. Packages must have only one level of Sub-Folders. Sub-Folders contain preset files. All preset files must be found directly within these Sub-Folders.

Packages may be expanded or collapsed. When collapsed, they do not show their Sub-Folder contents, and are not selected. Clicking on a collapsed Package will expand it to show all sub-folders. All Sub-Folders within the Package will be automatically selected as well, and add all of the preset files contained within these Sub-Folders to the File Browser results. Clicking on an expanded Package will collapse it, deselect all of its Sub-Folders, and thus remove all of the preset files contained within these Sub-Folders from the File Browser results.

The following actions are possible in the Package & Sub-Folder List:

- **Select Individual Packages:** Clicking on a single collapsed Package will automatically expand the selected Package, select all of its Sub-Folders, and display the Sub-Folder contents in the File Browser. Additionally it will collapse and de-select all other Packages, and remove all of the preset files contained within these Sub-Folders from the File Browser results.
- **Select Multiple Packages:** Shift-Clicking on multiple Packages will allow multiple Packages to be expanded and selected.
- **Select Individual Sub-Folders:** Clicking on a single Sub-Folder will automatically de-select all other Sub-Folders, select the current Sub-Folder, and display its contents in the File Browser.
- **Select Multiple Sub-Folders:** Shift-Clicking on multiple Sub-Folders will allow multiple Sub-Folders to be selected and display their contents in the File Browser.

Select All & None

Two additional selection tools are offered for Packages:

- **Select All:** Clicking on this button expands all Packages, selects all Sub-Folders, and displays all of their contents in the File Browser.
- **Select None:** Clicking on this button collapses all Packages, de-selects all Sub-Folders, and removes all preset files from the File Browser results.

The general idea of the Package and Sub-Folder scheme is to make it easier to find what you are looking for as well as to have an easy way to browse the contents of a newly purchased Preset Expansions.

File Browser

The Preset Browser File list displays the contents of the currently selected Sub-Folders. Selected Sub-Folder names are also included in the display area as well, and are clearly delineated from Preset names to achieve maximum organizational clarity. This makes it clear at a glance which folder a given preset belongs to. The active preset selection is highlighted by a different background color. File Browser results are also subjected to the current Keyword Search keywords.

Scrollbars, Increment & Decrement

To the left of the Package and Sub-Folder List and at the bottom of the File Browser are found scrollbar interface elements on an as-needed basis. If the currently selected contents exceed the available size of these areas to display them, scrollbars become available to allow you to scroll as needed to see the full contents available. Increment and Decrement buttons are also offered to shift the viewable contents one row or column at a time.

Keyword Search

Above the File Browser Kaleidoscope offers three text-entry fields that can be used for keyword searches. Kaleidoscope will search the preset contents of the currently displayed presets found in the File Browser and adjust its contents to only those presets that match the keyword search. Keyword search is applied only to selected Sub-Folders. If you would like to search across any and all presets, found in any Sub-Folder, from any Package, use the Select All button to be sure all presets are available for searching.

Keyword Boolean Mode

Three Keywords are offered for searching. These three Keywords can use either Boolean AND or Boolean OR logic to determine if a given preset is a match or not. The Boolean Mode switch determines which logic rule is used.

- **OR:** the left state of the Boolean Mode switch, if **ANY** of the keywords match, the search is a match
- **AND:** the right state of the Boolean Mode switch, if **ALL** of the keywords match, the search is a match

OR will produce more matches than AND. The GUI iconography uses a Venn Diagram to denote these two states.

Advanced Keyword Search

In previous 2CAudio products, Keyword Search was limited to the Preset File Name. In Kaleidoscope search functionality has been extended to include Preset Author Name as well as to search for particular parameter values and ranges. Search functionality is quite sophisticated. Special syntax is used for these type of advanced searches

- **To search for Author Name use the “@” symbol before your text String.**
 - Partial matches will be found.
 - Many hosts do not like “space” characters, so it is best to avoid them and try partial matches.
 - Example: to search for presets created by Andrew Souter search for “@Andrew” or “@Souter”
- **To search for Parameter Names and Parameter Values use the “\$” symbol before your text String**

- Parameter Names and Values must be an exact match.
- Parameter Names and Values are supplied in the appendix.
- Depending on the parameter, some Parameter Values are text Strings, and others are numeric values
- Numeric values are saved in floating point format. GUI values are displayed in percent. Therefore a value of 50% on the GUI is represented as 0.5 in a preset
- The following operators are supported for numeric values: =, <, >, <=, >=
- For String values use: =
- If a parameter String value String has a space character in it, use “_” in place of the space.
- Examples:
 - \$Feedback>0.6 : returns all presets with Feedback greater than 60%
 - \$ResMode=2_String : returns all presets that use the “2 String” resonator mode.

All three search methods can be combined into a single query using the three keyword fields. For example:

- AND(Groove , @Andrew , \$Feedback>0.6) : returns all presets that contain the word Groove in the file name, were created by someone named Andrew, and have a feedback greater than 60%.
- OR(@Simon , @Sascha) : returns all presets that were created by someone named Simon or Sascha

Random Preset Selection Tools

Two Random selection tools are offered to make random preset selections to encourage happy accidents and inspire creativity.

- **Random File:** selects a random file from within the currently displayed File Browser results, which comes from the same Sub-Folder as the currently selected Preset File.
- **Random Folder:** selects a random file from within the currently displayed File Browser results, which can come from and of the currently displayed Sub-Folders shown in the File Browser.

Note: Random selections are chosen only from within the current display results, which are a function of Sub-Folder selection in the Package and Sub-Folder List, as well as the action of any current Keyword Searches. This is different and arguably more powerful than the random functionality of the custom panels available on the Main Page of the GUI.

Author Name & Info Display

The Author Name and Info display shows the Author Name and Author Info for the currently selected preset. This is helpful to note for search functionality if you like the style of a particular preset designer and would like to find more of his or her work. It also encourages people to share presets and become involved in the community.

Info Page Details

Tuning Info

The Tuning Info Area displays a summary of all Tuning-Related settings, and any comments regarding the current Scale Tuning that happen to be saved into the file. This can be useful for users who would like to become more knowledgeable about the specifics of alternative tuning.

GUI Options, Preferences, Audio Options, and Quality & Efficiency

The GUI Options, Preferences, Audio Options, and Quality & Efficiency settings areas found the Info Page contain various settings that control the general look and behavior of the GUI as well as other miscellaneous behavior. These settings may be changed either by directly clicking on the text field button that displays the current setting, or by using the associated drop down menu to directly select the desired setting. Some changes to settings take place immediately; others require the plug-in to be reloaded to take effect, as described for each item below.

When using multiple instances of Kaleidoscope in a host project, changes to Options, Preferences, and Quality & Efficiency settings are global for all instances of Kaleidoscope; however, the host project file must be closed and reopened before all instances of Kaleidoscope will use the new settings. In other words, changing Preferences in a single instance of Kaleidoscope will change the settings for all instances in the project, but the project file must be closed and reloaded to automatically switch the settings of all of the other instances used in the project.

GUI Options

The GUI Options area of the Info Page contains options that control the general look and input response of the GUI as well as other miscellaneous behavior.

Skin

The Skin preference allows you to change the GUI Skin of the plug-in. GUI Skins are sometimes called Themes by other software developers, and represent all graphic files and associated layout resources for the plug-in. Changing the Skin preference allows you select from a list of predefined Skin options and offers the potential to completely change the visual appearance of the plug-in. Skin changes are effective only after reloading the plug-in or host project file. Kaleidoscope 1.0 ships with one Skin option:

- **Moonlight:** The default skin as described previously in this document.

Changes to the skin, when available, require reloading the plug-in to become active.

Note: Additional Kaleidoscope skins may become available in the future.

Color Scheme

Kaleidoscope offers three color schemes:

- **Yellow**

- Image Maps are displayed using the Red and Green color channels. Center becomes Yellow.
- Primary GUI object color is Yellow.
- GUI highlights are Blue.
- **Cyan (default)**
 - Image Maps are displayed using the Green and Blue color channels. Center becomes Cyan.
 - Primary GUI object color is Cyan.
 - GUI highlights are Red.
- **Magenta**
 - Image Maps are displayed using the Blue and Red color channels. Center becomes Magenta.
 - Primary GUI object color is Magenta.
 - GUI highlights are Green.

Changes to the Color Scheme are immediate and may be adjusted easily to suite your mood. Reloading the GUI is not needed.

Tip: It's OK to use magenta sometimes. It's our secret. We won't tell anyone. Real men wear pink sometimes. 😊

Image Display

Image Display controls how data is displayed in the Image Map area:

- **Post-Modulation:** Image Map data is shown after the effects of Mod Strength, Mod Curve, Mod Offset and Image Width. This best represents that real data as it is currently being used. (Default.)
- **Pre-Modulation:** Image Map data is shown before the effects of Mod Strength, Mod Curve, Mod Offset and Image Width. This best represents the data as it exists on the image file on disk. In some cases of extreme settings of these controls, this option can make it easier to see the details of the Image Map more clearly.

Show Values

The Show Values preference allows you to control how and when the numerical values of each knob are displayed. Changes to this preference take place immediately. This preference has three options:

- **Always:** Numerical values are displayed on all knobs at all times
- **Never:** Numerical values are never displayed on any knob unless the knob is in Text Entry mode. This mode is designed to force users to use their ears when adjusting parameters instead of relying on using their eyes, mind, and numerical affinity.
- **While Tracking:** Numerical values are displayed only on the knob that is actively being changed by the user. This mode provides a minimalist GUI aesthetic while still communicating the relevant numerical feedback when it is needed. (Default.)

Show Pop-Ups

The Show Pop-Ups preference allows you to control how and when the Value Tracker Pop-Ups for each control are displayed. Changes to this preference take place immediately. This preference has three options:

- **Off:** Value Tracker Pop-Ups are never displayed.
- **When Changed:** Value Tracker Pop-Ups are displayed while the associated control is being manually changed. (Default.)
- **On Mouse Over:** Value Tracker Pop-Ups are displayed upon mouse-over. A small time delay is used to avoid excessive pop-ups. The mouse cursor must be held over the control for a short while before the Pop-Up will open.

Preferences

The Preferences area of the Info Page contains preferences that miscellaneous behavior.

Knob Motion

The Knob Motion preference determines how to interoperate mouse input gestures when using the mouse to change a parameter knob on the GUI. Changes to this preference take place immediately. This preference has three options:

- **Linear:** Parameter values increase when mouse click-drag movement is up or to the right in a straight line. Parameter values decrease when mouse click-drag movement is down or to the left in a straight line. (Default.)
- **Circular:** Parameter values increase when mouse click-drag movement is in a clockwise circular arc around the knob. Parameter values decrease when mouse click-drag movement is in a counter-clockwise circular arc around the knob. This motion emulates the physical action of turning a real hardware knob, though it is not exactly the same experience and most users prefer to use the Linear motion.
- **Follow Host:** This mode attempts to determine the standard knob motion preference of the host application and use the same setting for Kaleidoscope.

Tail Reporting

The Tail Reporting preference determines whether or not Kaleidoscope attempts to communicate the actual realized decay time length (i.e. The Tail) to the host application. Some hosts feature CPU saving-techniques where processing is suspended when there is no active audio region passing sound into the plug-in inserts in the mixer. This can potentially create issues for plug-ins such as reverbs and delays that continue to produce output long after the input audio region has stopped playing. Most hosts are able to correctly handle these situations without additional effort on our behalf, and it is not necessary to use the “Report Tail” setting. Some hosts are not quite as advanced however, and for these hosts we offer the “Report Tail” setting to keep the tails from being muted when no audio regions are active. Changes to this preference take place immediately. This preference has two options:

- **Do Not Report Tail:** This mode does not attempt to report the tail length to the host, and is the default.
- **Report Tail:** This mode attempts to report the tail length to the host, to accommodate rare situations where the host does not do this automatically.

Tuning Display

The Tuning Display preference determines the frequency range used to show the Tuning function. It has three options:

- **Auto:** Automatically switches between 8 and 16 octaves depending on used resonator frequencies.
- **8 Octaves:** an 8 Octave Range from 32 to 8192 Hertz. This covers the range of almost all musical instrument pitches.
- **16 Octaves:** a 16 Octave Range from 1 to 65,536 Hertz. An extreme frequency range useful for more experimental sound-design and scientific uses of Kaleidoscope.

Note: maximum resonator frequency is limited to the smaller of 25% of the internal sample rate and 32,768 Hertz for Strings and FIR Mode. Maximum resonator frequency is limited to the smaller of 50% of the internal sample rate and 32,768 Hertz for Springs Mode. Minimum resonator frequency is 1.0 Hertz for Springs, and 0.125 Hertz for Strings and FIR. Resonators with frequencies outside of this range are made Inactive automatically.

Phase Lock

The Phase Lock preference determines how Kaleidoscope’s internal modulation timing clock is synched with the Host.

- **To Host:** Kaleidoscope's internal modulation timing clock is synched and phase locked with the Host in a sample accurate manner. When host playback is stopped, Kaleidoscope's internal clock is stopped. Modulation position is adjusted to always be perfectly in synch with the proper host position.
- **Internal:** Kaleidoscope's internal modulation timing clock free-running. It starts as soon as the plug-in instance is loaded and continues to run. It is not synched to the host.

Note: Some waveform editor hosts such as Sony Sound Forge do not supply timing information to Kaleidoscope; in such hosts it is required to use the Internal Phase Lock preference. The timing position will always be reset to zero each time Kaleidoscope is applied in such Offline hosts.

Recall Last

The Recall Last Preference determines what preset settings are loaded each time a new instance of the plug-in is loaded.

- **Yes:** The last settings used by the plug-in are remembered and recalled when loading the new instance. (Default.)
- **No:** The internal default values for all settings are loaded each time a new instance is loaded.

Recall last can be especially useful when using audio editor applications such as Sony Sound Forge, or Steinberg Wavelab where the normal workflow is to load the plug-in and apply it to a single stereo audio file and then close the plug-in automatically.

Audio Options

The Audio Options area of the Info Page contains preferences that control the gain staging and global limiter behavior as well as other miscellaneous items.

Global Gain

The Global Gain preference allows a gain factor of 0dB, -6dB, -12dB, -18dB, or -24dB to be applied to all instances of the plugin. We put a ton of work into getting consistent levels out of Kaleidoscope, and this is generally the case when using a pure white noise input such as when White is set to 100% or higher. However, due to the nature of high feedback, high resonance systems, output levels can be quite unpredictable when using highly tonal input signals.

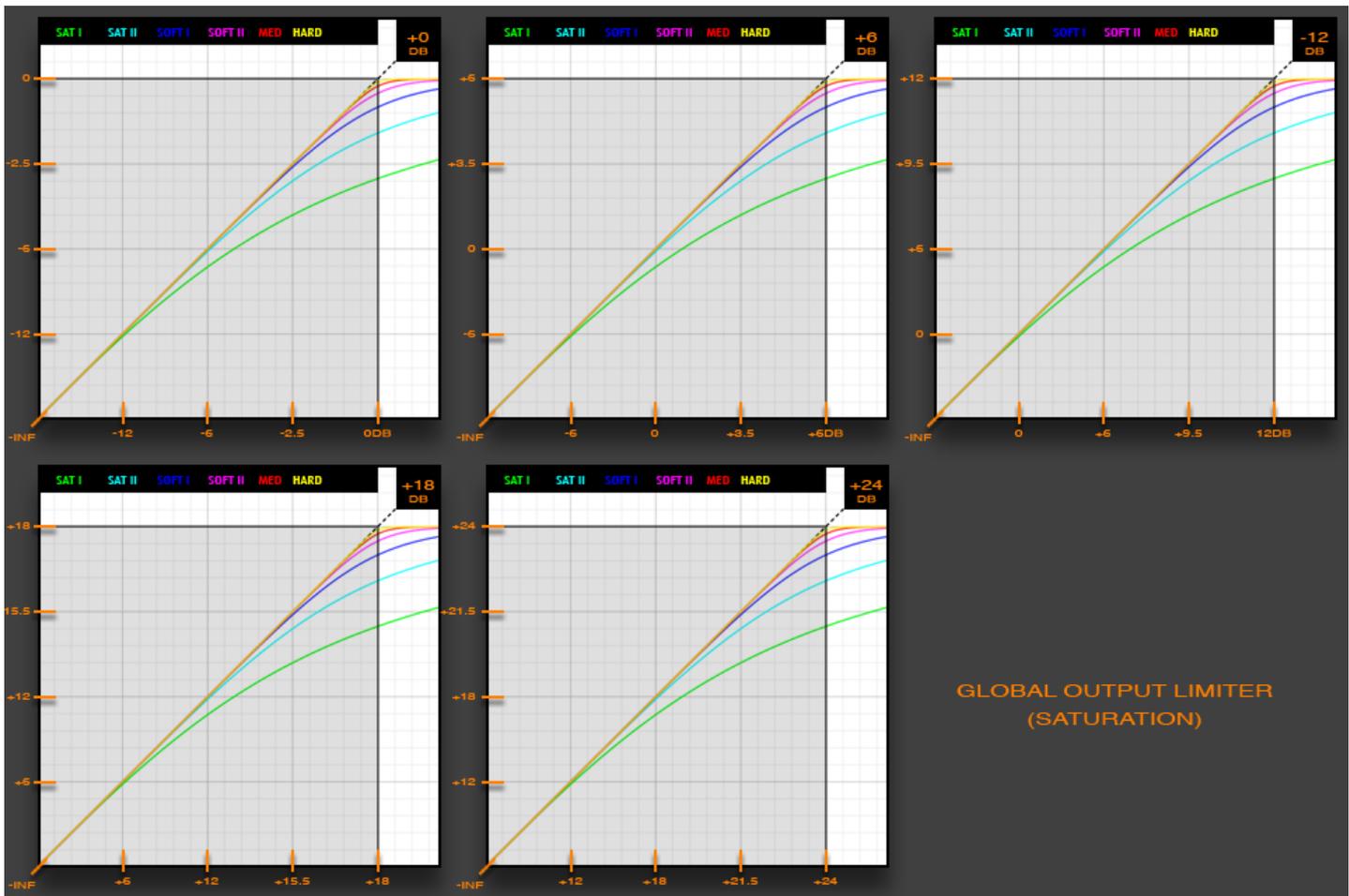
It is generally recommended to keep this setting at -12dB for use in all hosts. It can be useful however to use the -18dB or even the -24dB option when using single track audio editor applications such as Sony Sound Forge, or Steinberg Wavelab when applying to plug-in to fixed point audio files that are normalized to 0dB (such as sample loop libraries for example). This can prevent potential clipping potential in these situations.

-12dB will also provide the best results when using Kaleidoscope's built in white-noise generator by setting White to 200%. This should generate reasonable RMS levels, and peak levels approaching but not generally exceeding 0dB for almost all potential presets.

Limiter Mode

Kaleidoscope Features an optional Global output limiter that can be used to protect the output gain against going over the specified ceiling value. For extreme fidelity and the absolutely cleanest possible signal chain, the global limiter can be kept off. The limiter is not a clean, distortion free design. Instead it acts more like a saturation device that can and will add harmonic distortion to the signal as the signal level approaches the ceiling. The functionality and modes offered in the global limiter are in fact very similar to the first five Attitude types offered in our B2 product:

- **Sat I** A gentle Saturation function that adds a few odd harmonics
- **Sat II** A slightly less gentle Saturation function that adds a few odd harmonics
- **Soft I** A gentle Soft-Clip function that adds a few more odd harmonics
- **Soft II** A slightly less gentle Soft-Clip function that adds more odd harmonics
- **Med** A Medium knee clip function that adds many odd harmonics
- **Hard** A Hard-Clip function that adds an almost infinite number of odd harmonics and will alias



The graph above shows the behavior of the Global Limiter for all possible settings. The X-Axis represents the input, and the Y-Axis represents the output. The colored lines are the transfer function for the various Limiter Modes.

Softer curves are, for lack of a better word, softer. Softer curves will generate fewer high harmonics. They have a more gentle distortion characteristic. However, because of the wide/soft "knee" they will start the color the input signal at levels that are farther away (i.e. lower in gain) than the harder modes. The hard-clip mode for example will not really add any distortion right up to the ceiling level, but if this ceiling is exceeded, abrupt and fairly harsh clipping will occur which will contain many high frequency harmonics and will result in aliasing. If a softer model is selected, mild distortion to the signal may begin starting -6, -12, -18 or more dB down from the ceiling.

Limiter Ceiling

A nice trick to maintain some degree of level protection while minimizing coloration from harmonic distortion is to move the ceiling up into areas greater than 0dBFS to loosely emulate what might happen in the analog world. If we put the ceiling at +6dB or +12dB for example we can allow signals to go over 0dB, assuming we are working in a floating point environment

inside a modern DAW, but as the signal exceeds this level it will get progressively distorted and never exceed the specified ceiling. Within the nominal operating range of $-\infty$ to 0dBFS however, the curve will be closer to linear and thus it will have less distortion. Compare the numerical labels in the margins of the graphs above for each Limiter Ceiling choice, and note that higher the Ceiling is, the more linear the transfer function is in the nominal operating range from $-\infty$ to 0 dBFS.

+12dB with Sat I or Sat II is a nice starting point to try if you would like some gentle and subtle color and character that is reminiscent of the behavior of analog gear.

Due to the nature of resonance, unpredictable levels can be encountered in Kaleidoscope when using real musical signal inputs! It is highly advisable to either use Kaleidoscope's built-in Limiter, or to follow Kaleidoscope directly with a third party limiter in your multi-track DAW host of choice. Or do both.

Our recommendation for multi-track DAW use is to use a Global Gain of -12dB, together with a Limiter Ceiling of +12dB, using the Sat II Limiter mode. A third party limiter set to a 0dB ceiling should then follow Kaleidoscope. This will keep signals below 0dB almost perfectly clean, give some minor saturation character to any potential large transient spikes, and keep you completely protected against unwanted surprises.

Our recommendation for Offline stereo editor use is to use a Global Gain of -12dB, together with a Limiter Ceiling of +0dB, using the Soft II Limiter mode. This will keep signals below 0dB reasonably clean, give some minor saturation character to any potential large transient spikes, and keep you completely protected against unwanted surprises.

Reference Pitch

The Reference Pitch established a global tuning reference that used by Kaleidoscope to translate Note Tuning References to Hertz values. The current standard would be to use A=440 Hertz. However, you could easily use A=432 Hertz or C=256 Hertz or any other arbitrary standard if such things interest you. Other Note values are derived from this Global Reference Pitch assuming standard 12-Tone Equal Temperament. This is generally sufficient for 99.9% of all traditional musical needs. If you find it is not for your needs, you may use the "Frequency (Hertz)" Tuning Mode to directly enter Hertz-based tuning references on a per-preset basis.

The Reference Pitch Audio Option makes it easy to explore A=432 Hertz or C=256 Hertz and other tunings that are an active area of interest at the moment.

Default Tempo

The Default Tempo Audio Option is used by Kaleidoscope to establish tempo when presets are set to use the "Host (BPM & Ratio)" Timing Mode, and Kaleidoscope is being used in a host that does not supply Tempo information. In hosts such as Sound Forge, this makes it easy to set the tempo of the project you are working on and not have to worry about manually changing the tempo each time you load a new preset.

Quality Settings

Kaleidoscope offers some of the same advanced quality options that we introduced to the world in Aether 1.5. The features described below are designed to work synergistically to achieve a full frequency, perfectly band-limited, artifact-free system. The variable Quality Settings allow extreme levels of audio quality both for Real-Time use and Offline bounces. Quality Settings may be changed either by directly clicking on the text field button that displays the current setting, or by using the associated drop down menu to directly select the desired option. When using a single instance of Kaleidoscope, changes to Quality Settings take place immediately.

Note: When using multiple instances of Kaleidoscope in a host project, changes to Quality settings are global for all instances of Kaleidoscope; however, the host project file must be closed and reopened before all instances of Kaleidoscope will use the new Quality Settings. In other words, changing Quality Settings in a single instance of Kaleidoscope will change the settings for all instances in the project, but the project file must be closed and reloaded to switch the Quality Settings of all of the other instances used in the project.

Oversampling

Kaleidoscope allows 1X, 2X, and 4X oversampling options. Oversampling allows Kaleidoscope to internally run at a higher sample rate than the host sample rate. **Oversampling is most beneficial to the String Resonator modes.** In this mode it can increase filter quality, improve high frequency response, improve tuning accuracy in some cases, and reduce self-noise of the algorithm. Oversampling can also significantly increase the clarity in the high frequency response and help transient response. Furthermore, oversampling allows extremely flexible Damp filter and EQ settings to be used that would otherwise not be possible at normal base sample rates. Kaleidoscope's oversampling method is extremely high quality and does not add any artifacts or appreciable latency of its own, which we should note, is no easy task to achieve.

Oversampling has another benefit for Kaleidoscope's String resonators: String Resonator frequency cannot be higher than 25% of the sample rate. At a sample rate of 44,100 Hertz this means the highest possible resonator frequency is 11,025 Hertz. This is generally more than sufficient for Strings since Strings themselves contain harmonics and will extend the frequency response significantly higher, filling the entire spectrum. Almost all musical instruments have significantly lower maximum fundamental frequencies. The fundamental frequency of the highest A-note on a piano for example is just 3,520 Hertz. Using oversampling nonetheless allows String Resonator frequency to extend all the way to the Nyquist rate, 50% of the Sampling rate if desired.

Spring Resonator modes do not benefit from Oversampling nearly as much as the String modes do. Spring Resonator frequency can be set up to Nyquist rate, thus covering the entire audio spectrum without the use of Oversampling. Extreme high frequency settings may receive some minor benefit from Oversampling.

FIR Resonator mode receives almost no benefit whatsoever from Oversampling.

Our recommendation is to use 1X Oversampling (i.e. no Oversampling) for Real-Time and optionally use 2X or 4X for Offline bounces when and if you have the time to wait on the extended render times that will result from its use.

Max Resonators

The Max Resonators preference is a limit on the Max Enabled Lines slider on the Main page of the GUI. The Max Enabled Lines Slider cannot go higher than this value. This effectively limited to number of resonators used by presets in a global fashion. It can be used in cases of limited CPU power to limit how many resonators are used. It can be independently set for Real-Time and Offline use.

Multi-Threading

The Multi-Threading setting will determine how many threads Kaleidoscope uses to accomplish its processing workload. Each thread should in theory be assigned to a separate CPU core, but host applications and the operating system have influence on exactly how this is handled. The Multi-Threading value is a percentage of the number of logical cores that your computer has. Due to Hyper-Threading in modern CPUs, the number of logical cores is twice the number of physical cores in the system. A current, top-of-the-line 2014 Mac Pro has 12 physical cores, and 24 Logical cores for example. Using the 100% setting in Kaleidoscope would result in it using 24 processing threads in this case. 25% would result in 6 processing threads; 400% would result in 96 processing threads.

Using values greater than 100% can result in slightly greater efficiency in some cases according to our internal testing. Values less than 100% could be useful when using Kaleidoscope live in multi-track DAWs where another 3rd-party plug-in may be monopolizing all the CPU resources of a single core. In such cases it could be helpful to use fewer threads for Kaleidoscope.

The exact optimal setting of the Multi-Threading choice varies a lot based on the host, OS, and nature of the host project including what other plug-ins are being used simultaneously. The guidelines given below are just that, and we encourage you to try different settings to see what works best with your particular circumstances. Changes to this setting are effective immediately and do not require reloading the plug-in.

100% should be the default choice if you are unclear what to choose.

Buffer Size

Most audio signal processing processes process data in chunks or blocks of data. This is called a Buffer and is more or less the same thing as the Buffer size setting that is used for audio hardware devices and found in DAW setup preferences. Generally speaking the larger the Buffer size, the more efficient the computer is at processing the algorithm. This is perhaps even truer when dealing with heavily multi-threaded algorithms such as Kaleidoscope. The tradeoff of using large Buffer Size values is the additional latency that is a direct result. Kaleidoscope offers a variable Buffer size of 64, to 2048 samples. The larger the value is, the more efficient the plug-in will run; however the selected value is also the latency that will be reported to your host application. For pure mixing, and sound-design sessions, it is recommended to simply use the maximum size. For workflows where live performance is occurring, and the extra latency is objectionable, use a smaller size.

Our recommendation is use to the largest Buffer Size option whenever possible. This will increase efficiency very significantly. If the less latency is required for live performance needs, reduce the Buffer Size.

Real-Time & Offline Settings

The perennial "Performance vs. Quality" trade-off has now become obsolete! Kaleidoscope allows two independent sets of Quality Settings: one for Real-Time use, and one for Offline bounces. As such users can set Real-Time settings to a level appropriately matched to their project needs and available hardware power, while at the same time automatically invoking extreme quality standards during Offline bounces. Most modern host applications will communicate the processing state to plug-ins automatically and Kaleidoscope is able to intelligently switch between the two Quality Settings sets without additional input from the user.

There can be a tendency to "turn it to 11" with Kaleidoscope's Quality Settings, and admittedly the extreme settings can be interesting to explore since there is nothing else similar in the market at this time as far as we are aware. We should reiterate one last time however, that the lowest possible quality settings are already excellent. The most extreme settings really are just that, and we have designed them a little ahead of the current state of average available CPU power. In other words, we designed the extreme settings intentionally with some room to grow into them. The most extreme settings are simply too heavy for Real-Time use for the average CPU in 2014, but things change rapidly in technology and this may not be true in 12 to 18 months.

For your convenience we have prepared four different recommended Quality Settings guidelines based on available CPU power. Our current recommendation for an average 2014 computer is to use the Medium set of settings as shown below. If your available CPU power is limited, it is certainly not the end of the world to use the base quality settings. If it is higher, and you are as obsessive as we are, by all means please feel free to "turn-it-to-11" if you like. The beauty of these options is that they put the power directly in your hands and allow you to decide which settings are best for your needs. The following is provided as a guideline to help you make these decisions.

Force Offline

The Force Offline switch can use the Offline setting for Real-Time use. This can be useful for hosts that cannot automatically switch between Real-Time and Offline states. Kaleidoscope performs a few other undocumented tricks in Offline mode that are not available in Real-Time use, and this switch gives access to these in such circumstances as well.

CPU Power	Real-Time			Offline		
	Oversampling	Max Resonators	Multi-Threading	Oversampling	Max Resonators	Multi-Threading
Low	1x - Off	128	100%	1x - Off	512	300%
Medium	1x	256	100%	2x	512	300%
High	1x	512	100%	2x	512	300%
Extreme	1x	512	100%	4x	512	300%

User Name & User Info

User Name

The User Name field is used to tag user presets with the author name. Whenever a user preset is saved the User Name will be saved into the preset. When the preset is loaded, the name will be displayed in the Browser page to give credit to the author. This is useful when sharing presets with peers.

User Info

The User Info field is used to tag user presets with the author information. This information could be a web URL, a company name, a group name, a motto or slogan, or any other information you wish to include with your presets. It can be left blank as well. Whenever a user preset is saved the User Info will be saved into the preset. When the preset is loaded, the Info field will be displayed in the Browser page to give credit to the author. This is useful when sharing presets with peers.

Library Location

The Library Location field displays and allows you to change the location of the “2CAudio Resource Library” as described in the “Installation & Authorization” section of the “Introduction” chapter of this document.

Karma Boost Area

The Karma Boost area of the Info Page addresses product authorization. We call it Karma Boost because our motto here is “Do unto others as you would have done to yourself.” We do not employ authoritarian copy-protection schemes or hardware dongles. We prefer to treat our customers as our friends and make product authorization easy, non-invasive, and burden free. Our language choice in this area is designed to remind you that this sort of system is only possible with your support and reciprocation of our trust. We hope you will appreciate our decision and work with us to limit the distribution of pirated software. All we ask of you is this simple rule: “Do unto others as you would have done to yourself.”

Serial Number Field

The Serial Number Field is used to authorize the product as described in the “Installation & Authorization” section of the “Introduction” chapter of this document.

Version Number Display

The Version Number Display communicates the version of Kaleidoscope that is installed on your computer.

Authorization Status

The Authorization Status text field communicates exactly that. It can display several text Strings:

- **Enter Serial:** This is displayed before the product has been authorized
- **Authorized:** This is displayed when the product has been properly authorized and the plug-in is functioning correctly.

Additional text Strings are possible, but only occur when something is wrong. If encountered, please contact tech support.

Authorization Details

This area displays our friendly reminder regarding our philosophy on copy protection, and provides our blessing and appreciation for everyone who honors it.

Additional text Strings are possible, but only occur when something is wrong. If encountered, please contact tech support.

Appendix

File Formats

Image Formats

Images should be either:

- 24-bit PNG (8-bit per channel RGB, no Alpha transparency channel)
- 48-bit PNG (16-bit per channel RGB, no Alpha transparency channel)

Kaleidoscope used the Red and Green channel of these images. Red is used for the left audio channel, and Green is used for the right audio channel. Blue is disregarded completely. Kaleidoscope can remap these color channels to achieve different color schemes on the GUI, but new images should always be designed and saved to disk using the Red and Green channels.

Image size should be 1024x1024 pixels – even numbered pixels are used for up to 512 resonator lines. This allows for future expansion.

Scale Tuning Format

Scale Tuning Files are text files that loosely follow the Scala file format syntax developed by Manuel Op de Coul in the Netherlands. Those interested in alternative tuning theory and practice are encouraged to go here:

<http://www.huygens-fokker.org/scala/>

- The first line should start with a “!” and then list the file name.
- The second line should be only a “!” character.
- The third line can contain comments or be blank/empty.
- The fourth line should specify the number of Ratios in the file
- The fifth line should be only a “!” character
- The remaining lines should specify the tuning ratios for each line

Example:

```
!Semitones.txt
!
12-Tone Equal Temperament
12
!
1.000000
1.059463
1.122462
1.189207
1.259921
1.334839
1.414213
1.498307
1.587401
```

1.681792
 1.781797
 1.887748

Tuning ratios can be decimal in full double precision floating point or fractions with the numerator and denominator separated by a “/” symbol.

Waveform Tuning Format

Microsoft WAV format audio files are used to implement the tuning data. The files should be 64-bit floating point and Mono.

Waveforms must be 2048 samples long. Every 4th sample in the waveform is used to achieve the 512 samples tuning table. The Sample rate header does not matter.

Parameter Appendix

Following is a list of all parameter names and potential values. This is useful for advanced Keyword Searches.

-----	DryGain	
OUTPUT FORMAT:	DRY	[2]
index	[float]	White
str_id	[0.000000, 3.981072]	2
int_id	[0.000000]	White
full_str_id	[automatable]	WHITE
short name	[dB]	[float]
type		[0.000000, 2.000000]
value range	-----	[0.000000]
default value		[automatable]
automatable	[1]	[%]
label	WetGain	
int val String list	1	-----
-----	WetGain	
-----	WET	[3]
-----	[float]	BaseFreq
PLUG-IN EXPORTED PARAMS	[0.000000, 3.981072]	3
-----	[1.000000]	BaseFreq
	[automatable]	BSFREQ
[0]	[dB]	[double]
DryGain		[0.000000, 2000000000000000.000000]
0	-----	[47.000000]

[automatable]		
[Hz]	[7]	0: Musical Note
-----	ResMode	1: Hertz Frequency
	7	2: BPM Host
	ResMode	3: BPM Internal
[4]	RESMOD	4: Period
Feedback	[int32]	5: Wave Length
4	[0, 4]	6: Wav Frequency
Feedback	[0]	7: Wav Musical Note
FBK	[non automatable]	8: Wav Frequency Range
[float]	[]	-----
[0.000000, 1.000000]		
[0.500000]	0: 1 Spring	
[automatable]	1: 2 Spring	[10]
[]	2: 1 String	BaseFreqNoteOctave
-----	3: 2 String	10
	4: FIR	BaseFreqNoteOctave
	-----	FRQOCT
[5]		[int32]
MaxEnabledLines		[0, 15]
5	[8]	[6]
MaxEnabledLines	RandomDeTune	[non automatable]
MLINES	8	[]
[int32]	RandomDeTune	
[1, 512]	RNDDT	0: -5
[512]	[float]	1: -4
[non automatable]	[0.000000, 1.000000]	2: -3
[]	[0.500000]	3: -2
-----	[automatable]	4: -1
	[]	5: 0
	-----	6: 1
[6]		7: 2
PhaseWidth		8: 3
6	[9]	9: 4
PhaseWidth	FreqInputMode	10: 5
PHWDTH	9	11: 6
[float]	FreqInputMode	12: 7
[0.000000, 2.000000]	FREQMD	13: 8
[1.000000]	[int32]	14: 9
[automatable]	[0, 8]	15: 10
[]	[0]	-----
-----	[non automatable]	
	[]	

[11]	[13]	[0]
BaseFreqNoteIndex	ImageFile	[automatable]
11	13	[]
BaseFreqNoteIndex	MOD0-ImageFile	
FRQIDX	IMGFN	0: 1
[int32]	[String]	1: 2
[0, 11]	[Default Image]	2: 4
[9]	[non automatable]	3: 8
[non automatable]	[]	4: 16
[]		5: 32
	-----	6: 64
0: C		7: 128
1: C#	[14]	8: 256
2: D	ImageFilePath	9: 512
3: D#	14	10: 3
4: E	MOD0-ImageFilePath	11: 6
5: F	IMGFP	12: 12
6: F#	[String]	13: 24
7: G	[]	14: 48
8: G#	[non automatable]	15: 96
9: A	[]	16: 192
10: A#		17: 384
11: B	-----	18: 5
		19: 10
-----	[15]	20: 20
	ImageFileInfo	21: 40
[12]	15	22: 7
ModMode	MOD0-ImageFileInfo	23: 14
12	IMGFI	24: 28
MOD0-ModMode	[String]	25: 56
MODMD	[]	26: 9
[int32]	[non automatable]	27: 18
[0, 2]	[]	28: 36
[2]		29: 72
[non automatable]	-----	30: 15
[]		31: 30
	[16]	32: 60
0: Off	ModSyncNum	33: 120
1: Static	16	34: 11
2: Dynamic	MOD0-ModSyncNum	35: 13
	MODSN	36: 17
-----	[int32]	37: 19
	[0, 45]	38: 21

39: 23	22: 7	XOFFST
40: 25	23: 14	[int32]
41: 27	24: 28	[0, 1023]
42: 29	25: 56	[0]
43: 31	26: 9	[non automatable]
44: 33	27: 18	[]
45: 49	28: 36	
	29: 72	-----
	30: 15	
	31: 30	[20]
[17]	32: 60	ImageYOffset
ModSyncDenom	33: 120	20
17	34: 11	MOD0-ImageYOffset
MOD0-ModSyncDenom	35: 13	YOFFST
MODSD	36: 17	[int32]
[int32]	37: 19	[0, 511]
[0, 45]	38: 21	[0]
[0]	39: 23	[non automatable]
[automatable]	40: 25	[]
[]	41: 27	
	42: 29	-----
0: 1	43: 31	
1: 2	44: 33	[21]
2: 4	45: 49	ImageResizeY
3: 8		21
4: 16		MOD0-ImageResizeY
5: 32		IMGRSY
6: 64	[18]	[int32]
7: 128	ModSyncInfo	[0, 4]
8: 256	18	[0]
9: 512	MOD0-ModSyncInfo	[non automatable]
10: 3	MDSINF	[]
11: 6	[String]	
12: 12	[]	0: Off
13: 24	[non automatable]	1: 2X
14: 48	[]	2: 4X
15: 96		3: 8X
16: 192		4: 16X
17: 384		
18: 5	[19]	-----
19: 10	ImageXOffset	
20: 20	19	[22]
21: 40	MOD0-ImageXOffset	ModCurve

22	-----		-----
MOD0-ModCurve			
MODCV	[25]		
[int32]	ImageWidth	[28]	
[0, 6]	25	ModRate	
[0]	MOD0-ImageWidth	28	
[non automatable]	IMWDTH	MOD0-ModRate	
[]	[float]	MDRATE	
	[-1.000000, 1.000000]	[float]	
0: Power	[0.000000]	[0.000000, 1000000.000000]	
1: Exponential	[automatable]	[10.000000]	
2: CosinePower	[]	[non automatable]	
3: Arctan		[]	
4: Root	-----		
5: Bell		-----	
6: Stripes	[26]		
	TempoSync	[29]	
-----	26	ModRateNoteOctave	
	MOD0-TempoSync	29	
[23]	SYNC	MOD0-ModRateNoteOctave	
ModCvStrength	[int32]	MDRTNO	
23	[0, 3]	[int32]	
MOD0-ModCvStrength	[0]	[0, 15]	
MDCVST	[non automatable]	[6]	
[float]	[]	[non automatable]	
[-1.000000, 1.000000]		[]	
[0.000000]	0: Host		
[automatable]	1: Internal	0: -5	
[]	2: Free Time	1: -4	
	3: Musical AM	2: -3	
-----		3: -2	
	-----	4: -1	
[24]		5: 0	
ModDepthOffset	[27]	6: 1	
24	Tempo	7: 2	
MOD0-ModDepthOffset	27	8: 3	
MODDPS	MOD0-Tempo	9: 4	
[float]	TEMPO	10: 5	
[0.000000, 1.000000]	[float]	11: 6	
[0.000000]	[0.000000, 10000.000000]	12: 7	
[automatable]	[120.000000]	13: 8	
[]	[non automatable]	14: 9	
	[]	15: 10	

	32	
-----	MOD0-TempoInfo	-----
	TMPINF	
[30]	[float]	[35]
ModRateNoteIndex	[0.000000, 10000.000000]	ImageInversion
30	[120.000000]	35
MOD0-ModRateNoteIndex	[non automatable]	MOD0-ImageInversion
MDRTNI	[]	IMGINV
[int32]	-----	[int32]
[0, 11]		[0, 1]
[9]		[0]
[non automatable]	[33]	[non automatable]
[]	ModRateSec	[]
	33	
0: C	MOD0-ModRateSec	0: NoInvert
1: C#	MDRTD1	1: Invert
2: D	[float]	
3: D#	[0.000000, 1000000.000000]	-----
4: E	[10.000000]	
5: F	[non automatable]	[36]
6: F#	[]	XSize
7: G	-----	36
8: G#		MOD0-XSize
9: A		XSIZE
10: A#	[34]	[int32]
11: B	ImageOrientation	[1, 1024]
-----	34	[1024]
	MOD0-ImageOrientation	[non automatable]
[31]	IMGORN	[]
ModRateNoteInfo	[int32]	-----
31	[0, 7]	
MOD0-ModRateNoteInfo	[0]	
MDRNIN	[non automatable]	[37]
[String]	[]	ScanMode
[]	0: Std	37
[non automatable]	1: FlipX	MOD0-ScanMode
[]	2: FlipY	SCMODE
-----	3: FlipXY	[int32]
	4: Rotate	[0, 1]
	5: Rotate Flip X	[0]
[32]	6: Rotate Flip Y	[non automatable]
TempoInfo	7: Rotate Flip XY	[]

0: Equal Period	[String]	1: 2
1: Equal Rate	[Default Image]	2: 4
	[non automatable]	3: 8
-----	[]	4: 16
		5: 32
[38]	-----	6: 64
RandomOffset		7: 128
38	[41]	8: 256
MOD0-RandomOffset	ImageFilePath	9: 512
RNDOFS	41	10: 3
[int32]	MOD1-ImageFilePath	11: 6
[0, 1]	IMGFP	12: 12
[0]	[String]	13: 24
[non automatable]	[]	14: 48
[]	[non automatable]	15: 96
	[]	16: 192
0: None		17: 384
1: PreDelay	-----	18: 5
		19: 10
-----	[42]	20: 20
	ImageFileInfo	21: 40
[39]	42	22: 7
ModMode	MOD1-ImageFileInfo	23: 14
39	IMGFI	24: 28
MOD1-ModMode	[String]	25: 56
MODMD	[]	26: 9
[int32]	[non automatable]	27: 18
[0, 2]	[]	28: 36
[2]		29: 72
[non automatable]	-----	30: 15
[]		31: 30
	[43]	32: 60
0: Off	ModSyncNum	33: 120
1: Static	43	34: 11
2: Dynamic	MOD1-ModSyncNum	35: 13
	MODSN	36: 17
-----	[int32]	37: 19
	[0, 45]	38: 21
[40]	[0]	39: 23
ImageFile	[automatable]	40: 25
40	[]	41: 27
MOD1-ImageFile		42: 29
IMGFN	0: 1	43: 31

44: 33	27: 18	[]
45: 49	28: 36	
-----	29: 72	-----
	30: 15	
[44]	31: 30	[47]
ModSyncDenom	32: 60	ImageYOffset
44	33: 120	47
MOD1-ModSyncDenom	34: 11	MOD1-ImageYOffset
MODSD	35: 13	YOFFST
[int32]	36: 17	[int32]
[0, 45]	37: 19	[0, 511]
[0]	38: 21	[0]
[automatable]	39: 23	[non automatable]
[]	40: 25	[]
	41: 27	-----
0: 1	42: 29	
1: 2	43: 31	[48]
2: 4	44: 33	ImageResizeY
3: 8	45: 49	48
4: 16	-----	MOD1-ImageResizeY
5: 32		IMGRSY
6: 64	[45]	[int32]
7: 128	ModSyncInfo	[0, 4]
8: 256	45	[0]
9: 512	MOD1-ModSyncInfo	[non automatable]
10: 3	MDSINF	[]
11: 6	[String]	
12: 12	[]	0: Off
13: 24	[non automatable]	1: 2X
14: 48	[]	2: 4X
15: 96	-----	3: 8X
16: 192		4: 16X
17: 384		-----
18: 5	[46]	
19: 10	ImageXOffset	
20: 20	46	[49]
21: 40	MOD1-ImageXOffset	ModCurve
22: 7	XOFFST	49
23: 14	[int32]	MOD1-ModCurve
24: 28	[0, 1023]	MODCV
25: 56	[0]	[int32]
26: 9	[non automatable]	[0, 6]

[0]	MOD1-ImageWidth	55
[non automatable]	IMWDTH	MOD1-ModRate
[]	[float]	MDRATE
	[-1.000000, 1.000000]	[float]
0: Power	[0.000000]	[0.000000, 1000000.000000]
1: Exponential	[automatable]	[10.000000]
2: CosinePower	[]	[non automatable]
3: Arctan		[]
4: Root	-----	-----
5: Bell		-----
6: Stripes	[53]	
-----	TempoSync	[56]
	53	ModRateNoteOctave
	MOD1-TempoSync	56
[50]	SYNC	MOD1-ModRateNoteOctave
ModCvStrength	[int32]	MDRTNO
50	[0, 3]	[int32]
MOD1-ModCvStrength	[0]	[0, 15]
MDCVST	[non automatable]	[6]
[float]	[]	[non automatable]
[-1.000000, 1.000000]		[]
[0.000000]	0: Host	
[automatable]	1: Internal	0: -5
[]	2: Free Time	1: -4
-----	3: Musical AM	2: -3
	-----	3: -2
		4: -1
[51]		5: 0
ModDepthOffset	[54]	6: 1
51	Tempo	7: 2
MOD1-ModDepthOffset	54	8: 3
MODDPS	MOD1-Tempo	9: 4
[float]	TEMPO	10: 5
[0.000000, 1.000000]	[float]	11: 6
[0.000000]	[0.000000, 10000.000000]	12: 7
[automatable]	[120.000000]	13: 8
[]	[non automatable]	14: 9
-----	[]	15: 10
	-----	-----
[52]		
ImageWidth	[55]	[57]
52	ModRate	ModRateNoteIndex

57	[120.000000]	62
MOD1-ModRateNoteIndex	[non automatable]	MOD1-ImageInversion
MDRTNI	[]	IMGINV
[int32]	-----	[int32]
[0, 11]		[0, 1]
[9]		[0]
[non automatable]	[60]	[non automatable]
[]	ModRateSec	[]
	60	
0: C	MOD1-ModRateSec	0: NoInvert
1: C#	MDRTD1	1: Invert
2: D	[float]	
3: D#	[0.000000, 1000000.000000]	-----
4: E	[10.000000]	
5: F	[non automatable]	[63]
6: F#	[]	XSize
7: G		63
8: G#	-----	MOD1-XSize
9: A		XSIZE
10: A#	[61]	[int32]
11: B	ImageOrientation	[1, 1024]
	61	[1024]
-----	MOD1-ImageOrientation	[non automatable]
	IMGORN	[]
[58]	[int32]	-----
ModRateNoteInfo	[0, 7]	
58	[0]	
MOD1-ModRateNoteInfo	[non automatable]	[64]
MDRNIN	[]	ScanMode
[String]		64
[]	0: Std	MOD1-ScanMode
[non automatable]	1: FlipX	SCMODE
[]	2: FlipY	[int32]
-----	3: FlipXY	[0, 1]
	4: Rotate	[0]
	5: Rotate Flip X	[non automatable]
	6: Rotate Flip Y	[]
[59]	7: Rotate Flip XY	
TempoInfo		0: Equal Period
59	-----	1: Equal Rate
MOD1-TempoInfo		
TMPINF		-----
[float]	[62]	
[0.000000, 10000.000000]	ImageInversion	

0: Fixed	[]	FTLOWT
1: Relative		[int32]
	-----	[0, 3]
		[0]
	[77]	[automatable]
[74]	SoftFunction	[]
RandomDistribution	77	
74	SoftFunction	0: 6 dB
RandomDistribution	SFTFNC	1: 12 dB
RNDDSR	[int32]	2: 24 dB
[int32]	[0, 4]	3: Off
[0, 1]	[0]	
[0]	[automatable]	-----
[automatable]	[]	
[]		[80]
	0: Harmonic	FilterLowRouting
0: Gaussian	1: Exponential	80
1: Uniform	2: Linear	FilterLowRouting
	3: Butterworth	FTLOWR
	4: AsymBell	[int32]
		[0, 2]
	-----	[1]
[75]		[non automatable]
RandomPhase	[78]	[]
75	SoftCurve	
RandomPhase	78	0: In
RNDPH	SoftCurve	1: Out
[float]	SFTCV	2: InOut
[0.000000, 2.000000]	[int32]	
[1.000000]	[0, 1]	-----
[automatable]	[0]	
[]	[automatable]	[81]
	[]	FilterLowFreq
		81
	0: Linear	FilterLowFreq
[76]	1: Squared	FTLOWF
Soft		[float]
76	-----	[8.000000, 32768.000000]
Soft		[64.000000]
SOFT	[79]	[automatable]
[float]	FilterLowType	[]
[0.000000, 1.000000]	79	-----
[0.500000]	FilterLowType	
[automatable]		

	[4096.000000]	26: 9
[82]	[automatable]	27: 18
FilterHighType	[]	28: 36
82		29: 72
FilterHighType	-----	30: 15
FTHIT		31: 30
[int32]	[85]	32: 60
[0, 3]	TuningBpmSyncNum	33: 120
[0]	85	34: 11
[automatable]	TuningBpmSyncNum	35: 13
[]	TNBSNM	36: 17
	[int32]	37: 19
0: 6 dB	[0, 45]	38: 21
1: 12 dB	[0]	39: 23
2: 24 dB	[automatable]	40: 25
3: Off	[]	41: 27
		42: 29
-----	0: 1	43: 31
	1: 2	44: 33
[83]	2: 4	45: 49
FilterHighRouting	3: 8	
83	4: 16	-----
FilterHighRouting	5: 32	
FTHIR	6: 64	[86]
[int32]	7: 128	TuningBpmSyncDenom
[0, 2]	8: 256	86
[1]	9: 512	TuningBpmSyncDenom
[non automatable]	10: 3	TNBSDN
[]	11: 6	[int32]
	12: 12	[0, 45]
0: In	13: 24	[0]
1: Out	14: 48	[automatable]
2: InOut	15: 96	[]
	16: 192	
-----	17: 384	0: 1
	18: 5	1: 2
[84]	19: 10	2: 4
FilterHighFreq	20: 20	3: 8
84	21: 40	4: 16
FilterHighFreq	22: 7	5: 32
FTHIF	23: 14	6: 64
[float]	24: 28	7: 128
[8.000000, 32768.000000]	25: 56	8: 256

9: 512	TuningTempInt	[]
10: 3	TNTMIN	
11: 6	[float]	0: 1
12: 12	[0.000000, 10000.000000]	1: 2
13: 24	[120.000000]	2: 4
14: 48	[non automatable]	3: 8
15: 96	[]	4: 16
16: 192		5: 32
17: 384	-----	6: 64
18: 5		
19: 10	[88]	-----
20: 20	TuningScaleFile	
21: 40	88	[91]
22: 7	TuningScaleFile	TuningPart
23: 14	SCALE	91
24: 28	[String]	TuningPart
25: 56	[Default Scale]	TUNPRT
26: 9	[non automatable]	[int32]
27: 18	[]	[0, 6]
28: 36		[0]
29: 72	-----	[non automatable]
30: 15		[]
31: 30	[89]	
32: 60	TuningScaleFilePath	0: 1
33: 120	89	1: 2
34: 11	TuningScaleFilePath	2: 4
35: 13	SCPATH	3: 8
36: 17	[String]	4: 16
37: 19	[]	5: 32
38: 21	[non automatable]	6: 64
39: 23	[]	
40: 25		-----
41: 27	-----	
42: 29		[92]
43: 31	[90]	TuningSort
44: 33	TuningDup	92
45: 49	90	TuningSort
	TuningDup	TUNSRRT
-----	TUNDUP	[int32]
	[int32]	[0, 2]
[87]	[0, 6]	[0]
TuningTempInt	[0]	[non automatable]
87	[non automatable]	[]

0: Off	[]	[0, 1]
1: Ascending	-----	[0]
2: Descending		[automatable]
		[]
	[96]	
-----	TuningWavRange	0: Input
	96	1: Output
	TuningWavRange	
[93]	TUNWVR	-----
TuningOctaves	[double]	
93	[0.000000, 2000000000000000.000000]	[99]
TuningOctaves	[1000.000000]	NumDefinedLines
TUNOCT	[automatable]	102
[int32]	[]	NumDefinedLines
[0, 1]	-----	LNDEF
[0]		[int32]
[non automatable]		[0, 512]
[]	[97]	[512]
0: No Repeat	TuningWavSpace	[non automatable]
1: Repeat	97	[]
	TuningWavSpace	-----
-----	TUNWVS	
	[int32]	[100]
[94]	[0, 1]	NumEnabledLines
TuningWavFile	[0]	101
94	[non automatable]	NumEnabledLines
TuningWavFile	[]	LNENAB
TUNWAV		[int32]
[String]	0: Log	[0, 512]
[Default Waveform]	1: Lin	[512]
[non automatable]	-----	[non automatable]
[]	-----	[]
-----	INTERNAL PARAMS	-----
[95]		[101]
TuningWavFilePath	[98]	NumActiveLines
95	ModSiteView	100
TuningWavFilePath	99	NumActiveLines
WAVPTH	ModSiteView	LNACTV
[String]	MSVIEW	[int32]
[]	[int32]	[0, 512]
[non automatable]		

[1]	TuningBpmSyncInfo	ShowTuneMap
[non automatable]	106	SHOTUN
[]	TuningBpmSyncInfo	[int32]
-----	TNBSIN	[0, 1]
	[String]	[1]
	[]	[automatable]
[102]	[non automatable]	[]
TuningFileComment	[]	
103	-----	0: Hide
TuningFileComment		1: Show
TUNCOM		
[String]	[106]	-----
[]	HostTempo	
[non automatable]	107	[109]
[]	HostTempo	ShowXGrid
-----	HSTTMP	110
	[float]	ShowXGrid
	[0.000000, 10000.000000]	SHOXGR
[103]	[120.000000]	[int32]
TuningScaleFileInfo	[non automatable]	[0, 1]
104	[]	[1]
TuningScaleFileInfo	-----	[automatable]
TUNINF		[]
[String]	[107]	0: Hide
[]	ShowModImage	1: Show
[non automatable]	108	
[]	ShowModImage	-----
-----	SHOIMG	
	[int32]	[110]
[104]	[0, 1]	ShowYGrid
TuningWavFileInfo	[1]	111
105	[automatable]	ShowYGrid
TuningWavFileInfo	[]	SHOYGR
TNWFIN		[int32]
[String]	0: Hide	[0, 1]
[]	1: Show	[1]
[non automatable]	-----	[automatable]
[]		[]
-----	[108]	0: Hide
	ShowTuneMap	1: Show
[105]	109	

-----	-----	□
[111]	[114]	-----
XGrid	ImageCtlModCvView	
112	115	[117]
XGrid	ImageCtlModCvView	TuningSettingsInfo
XGRID	IMVMCV	118
[int32]	[int32]	TuningSettingsInfo
[1, 1024]	[0, 1]	TNSTIN
[128]	[0]	[String]
[non automatable]	[non automatable]	□
□	□	[non automatable]
-----	0: Pre-Modulation	□
	1: Post-Modulation	-----
[112]	-----	[118]
YGrid		RtXS
113		119
YGrid	[115]	RtXS
YGRID	TuningMapRange	XSRT
[int32]	116	[int32]
[1, 512]	TuningMapRange	[0, 2]
[128]	TNMPRG	[0]
[non automatable]	[int32]	[non automatable]
□	[0, 2]	□
-----	[0]	0: 1X
	[non automatable]	1: 2X
[113]	□	2: 4X
ImageViewColorScheme	0: Auto	-----
114	1: 8	
ImageViewColorScheme	2: 16	[119]
IMGVCS	-----	RtMaxEnabledLinesLimit
[int32]		120
[0, 2]	[116]	RtMaxEnabledLinesLimit
[0]	FreqNoteInfo	ENLLRT
[non automatable]	117	[int32]
□	FreqNoteInfo	[0, 3]
0: RGB	FQNTIN	[2]
1: GBR	[String]	[non automatable]
2: BRG	□	□
	[non automatable]	

	[122]	TNLOCK
0: 64	RefNoteIndex	[int32]
1: 128	123	[0, 3]
2: 256	RefNoteIndex	[0]
3: 512	NOTIDX	[non automatable]
-----	[int32]	[]
	[0, 11]	
	[9]	0: None
[120]	[non automatable]	1: Note
OffiXS	[]	2: RefFreq
121		3: All
OffiXS	0: C	-----
XSOL	1: C#	
[int32]	2: D	[125]
[0, 2]	3: D#	CircleYLock
[2]	4: E	126
[non automatable]	5: F	CircleYLock
[]	6: F#	CRCLK
	7: G	[int32]
0: 1X	8: G#	[0, 1]
1: 2X	9: A	[1]
2: 4X	10: A#	[non automatable]
-----	11: B	[]
	-----	-----
[121]		
OffiMaxEnabledLinesLimit	[123]	
122	RefNoteFreq	[126]
OffiMaxEnabledLinesLimit	124	0
ENLLOL	RefNoteFreq	DryGain
[int32]	REFFRQ	[float]
[0, 3]	[double]	[0.000000, 1.000000]
[2]	[1.000000, 10000.000000]	[0.000000]
[non automatable]	[440.000000]	[non automatable]
[]	[automatable]	[dB]
	[Hz]	
0: 64	-----	-----
1: 128		
2: 256		
3: 512	[124]	
-----	TuningLock	[127]
	125	SetupAB
	TuningLock	100000

SetupAB	[]	100007
SetAB		PresetAuthorInfo
[int32]	-----	
[0, 1]		[String]
[0]	[131]	[]
[non automatable]	PresetCategory	[non automatable]
[]	100004	[]
	PresetCategory	-----
	[String]	
[128]	[]	[135]
ActiveView	[non automatable]	PresetAuthorGuiInfo
100001	[]	100008
ActiveView		PresetAuthorGuiInfo
View	-----	
[int32]		[String]
[0, 10]	[132]	[]
[0]	PresetOrigin	[non automatable]
[non automatable]	100005	[]
[]	PresetOrigin	-----
	[int32]	
	[0, 3]	[136]
[129]	[0]	PrefPresetAuthorName
PresetInfo	[non automatable]	100009
100002	[]	PrefPresetAuthorName
PresetInfo	-----	
[String]		[String]
[]	[133]	[]
[non automatable]	PresetAuthorName	[non automatable]
[]	100006	[]
	PresetAuthorName	-----
	[String]	
[130]	[]	[137]
PresetName	[non automatable]	PrefPresetAuthorInfo
100003	[]	100010
PresetName		PrefPresetAuthorInfo

[String]		[String]
[]	[134]	[]
[non automatable]	PresetAuthorInfo	[non automatable]
		[]

	PBFolderListSBPos	
	PBFISP	
	[int32]	
[138]	[0, 1000]	[145]
PresetShortDescription	[0]	PBSearchKeywordMode
100011	[non automatable]	100018
PresetShortDescription	[]	PBSearchKeywordMode
[String]		[int32]
[]		[0, 1]
[non automatable]	[142]	[0]
[]	PBSearchKeyword1	[non automatable]
	100015	[]
	PBSearchKeyword1	
		0: OR
[139]	[String]	1: AND
ProdVersion	[]	
100012	[non automatable]	
ProdVersion	[]	
ProdVersion		[146]
[String]		SkinNameToLoad
[]		100019
[non automatable]	[143]	SkinNameToLoad
[]	PBSearchKeyword2	
	100016	[int32]
	PBSearchKeyword2	[0, 1]
		[0]
[140]	[String]	[non automatable]
PBColShift	[]	[]
100013	[non automatable]	
PBColShift	[]	0: Moonlight
PBCISh		1: Test_bkp
[int32]		
[0, 100]		
[0]	[144]	
[non automatable]	PBSearchKeyword3	[147]
[]	100017	KnobShowValMode
	PBSearchKeyword3	100020
		KnobShowValMode
	[String]	
[141]	[]	[int32]
PBFolderListSBPos	[non automatable]	[0, 2]
100014	[]	[0]

[non automatable]	OfflineMode	3: 18 dB
[]	OfIMod	4: 24 dB
0: Always	[int32]	-----
1: Never	[0, 1]	
2: While Tracking	[0]	
-----	[non automatable]	[153]
	[]	OutputLimiterClipMode
	0: Auto	100026
	1: Force	OutputLimiterClipMode
[148]	-----	OutLimClip
KnobTrackMode		[int32]
100021		[0, 6]
KnobTrackMode		[3]
	[151]	[non automatable]
[int32]	ReportTail	[]
[0, 0]	100024	
[0]	ReportTail	0: Off
[non automatable]	RTail	1: Sat I
[]	[int32]	2: Sat II
0: Linear	[0, 1]	3: Soft I
-----	[0]	4: Soft II
	[non automatable]	5: Med
	[]	6: Hard
	0: No	-----
[149]	1: Yes	
LoadLastPreset	-----	[154]
100022		GlobalGainAdjustment
LoadLastPreset		100027
	[152]	GlobalGainAdjustment
[int32]	OutputLimiterThreshold	GLGNAD
[0, 1]	100025	[int32]
[0]	OutputLimiterThreshold	[0, 4]
[non automatable]	OutLimTh	[0]
[]	[int32]	[non automatable]
0: No	[0, 4]	[]
1: Yes	[2]	
-----	[non automatable]	0: 0 dB
	[]	1: -6 dB
		2: -12 dB
[150]	0: 0 dB	3: -18 dB
OfflineMode	1: 6 dB	4: -24 dB
100023	2: 12 dB	

-----	DPMCNO	
[155]	[int32]	0: host
DspProcMultiCoreNumOptRt	[0, 6]	1: internal
100028	[6]	
DspProcMultiCoreNumOptRt	[non automatable]	-----
DPMCNR	[]	[160]
[int32]	0: use 1	DefaultTempo
[0, 6]	1: use 2	100033
[6]	2: use 4	DefaultTempo
[non automatable]	3: use 8	DEFTMP
[]	4: save 2	[float]
	5: save 1	[20.000000, 300.000000]
0: use 1	6: use all	[120.000000]
1: use 2		[automatable]
2: use 4	-----	[BPM]
3: use 8		
4: save 2	[158]	-----
5: save 1	DspProcMultiCoreAllocTypeOffl	
6: use all	100031	[161]
	DspProcMultiCoreAllocTypeOffl	ValueTrackerShowMode
-----	DPMCAO	100034
[156]	[int32]	ValueTrackerShowMode
DspProcMultiCoreAllocTypeRt	[0, 1]	VTSHMD
100029	[0]	[int32]
DspProcMultiCoreAllocTypeRt	[non automatable]	[0, 2]
DPMCAR	[]	[1]
[int32]	0: per physical core	[non automatable]
[0, 1]	1: per virtual core	[]
[0]		0: Off
[non automatable]	-----	1: OnValChange
[]		2: OnMouseover
0: per physical core	[159]	
1: per virtual core	PhaseLock	-----
	100032	
	PhaseLock	[162]
-----	PHSLCK	SharedContentFolder
[157]	[int32]	100035
DspProcMultiCoreNumOptOffl	[0, 1]	SharedContentFolder
100030	[0]	SHCTFL
DspProcMultiCoreNumOptOffl	[non automatable]	[String]
	[]	[]

[non automatable]	100039
[]	AuthPrompt

	[String]
	[]
[163]	[non automatable]
SharedContentFolderBtnInfo	[]
100036	
SharedContentFolderBtnInfo	
SHCTBI	
[String]	
[[click here to locate 2CAudio Resource Library]]	
[non automatable]	
[]	

[164]	
AuthSnInfo	
100037	
AuthSnInfo	
[String]	
[]	
[non automatable]	
[]	

[165]	
AuthStatus	
100038	
AuthStatus	
[String]	
[]	
[non automatable]	
[]	

[166]	
AuthPrompt	