# I CHOP Editor

## I INTRODUCTION

CHOPs (Channel Operators) are a powerful technology which enables the editing of motion and audio using procedural networks.

- *Editing Motion using Procedural Networks* – Motion can be edited and refined using procedural operators. CHOPs make the assembly, refinement and management of motion and audio radically easier.
- *Audio Synthesis Integrated with 3D Animation* – Audio can be imported, synthesized and edited with CHOPs, and can be driven by the motion of 3D objects.

CHOPs provide a new way of creating and editing motion and audio. The unprecedented combination of procedural editing of motion, keyframe animation, motion capture and live puppeteering provides you with the deepest range of animation techniques.

CHOPs fit seamlessly into the Houdini user interface as another editor. A CHOP contains one or more motion curves or "channels". CHOPs are connected to each other in a network. A CHOP modifies the channels, and then passes the channels on to the next CHOP in the network. CHOPs are then connected to object motion or any other animatable part of Houdini.

In keeping with the Houdini procedural paradigm, channel operators combine and refine the motion without destroying the original data. This lets you experiment with motion more than any other animation technology.

CHOPs were designed to reduce the tedium of motion editing, to help build and manage more complex motion, and to enable you to think about motion at a more creative level. We hope that as you use CHOPs, this becomes increasingly apparent.

To give you some ideas of how CHOPs can be applied to your work, here is a description of the major uses of CHOPs.

## 1.1 USES OF CHOPS

### NON-DESTRUCTIVE MOTION EDITING

You can edit motion using CHOPs in a way that doesn't affect the original motion channels. You can change the timing of motion by shifting, compressing, expanding and warping time. You can apply operators that smooth, reverse, repeat, trim and extend motion. You can go back to any CHOP, change its effect on the resulting motion, and experiment with modified timing easily.

### MOTION COMPOSITING

A group of CHOPs, each containing a motion clip for a character, can be sequenced one after the other, with controls over the blending transition between the motion clips. You can start with one motion clip and temporarily override it with another motion clip. Motion can be layered so that gestures can temporarily and partially override other gestures. Because you can experiment on the blending without destroying the original data, you can easily fine tune layered motion.

### FACIAL ANIMATION

Facial motion can be animated from a set of facial expressions, made of poses and motion clips. Facial poses and motion clips can be composed from other poses and facial motion clips and then re-used.

### LIP SYNC

Lip sync is the facial animation task that uses a voice soundtrack and a library of mouth poses to animate a character's mouth. You can import a voice track, add words of the dialogue to the time line, drop mouth poses and gestures on the time line, then adjust the timing of the poses. You can also filter and smooth the audio to automatically generate approximate mouth motion.

### MOTION CAPTURE EDITING

You can import motion capture clips from popular systems such as Acclaim, Biovision and Motion Analysis. Defects in captured motion can be cleaned up. Motion capture data can be applied to an inverse or forward kinematics character model, and you can further refine the motion with CHOPs. A CHOP holds multiple-channels of motion. You can easily attach and detach different motion capture moves to one character.

## REAL-TIME PUPPETEERING

You can attach the mouse, MIDI or other devices to objects one at a time and perform the motion of the object, building up the motion of a character or object, piece by piece. You can perform these "gestures" with the mouse or other devices and capture the motion in a CHOP. You can edit the captured motion immediately by trimming, shifting, smoothing, and then combining it with other motion clips. Houdini's real-time puppeteering (known in Houdini as "BeatBox") is a unique way of rapidly creating motion.

## KEYFRAME POSE INTERPOLATION

CHOPs provides an alternative way to hand-edit keyframing in Houdini, aside from the channel keyframing documented in the User Guide.

Within CHOPs you can pose a character, snapshot the angles and IK end-effectors into a single CHOP, sequence several poses one after another, then adjust a timing curve in another CHOP to interpolate between poses. You can easily edit the poses and timing, and you can copy-paste poses and animation to other characters.

## AUDIO EDITING

You can import audio into Houdini from external sources and then trim, filter, resample, sync and mix the sound. You can attach mixed audio to the animation timeline. The 3D animation can be synchronized with the soundtrack with the help of timeline scrubbing and text annotation on the timeline.

## AUDIO SYNTHESIS AND 3D SOUND

Sound effects can be imported from external sources, but audio can also be synthesized from scratch using built-in audio oscillators, filters, reverb and volume controls. Sound effects can then be triggered by any event or motion in the 3D scene, or a sound can be continuously looped. Then the audio can be placed anywhere in the 3D environment of the animation, with audio cues like left-right pan, distance attenuation, muffling and doppler shift. Sounds can be triggered to help users understand conditions in the 3D scene, such as character joint limits. Audio is generated in real-time, or more complex audio can be "rendered" and played back with the animation.

## DYNAMICS AND SPECIAL EFFECTS MOTION

The creation of special motion effects is easier with CHOPs. You can detect events in the 3D scene, like collisions or direction changes, and then trigger the start of other motion. CHOPs include many tools to smooth curves, simulate physical motion, and add natural-looking random variations to motion. For example, CHOPs can be assembled to simulate physical camera motion, including lags, shakes, bumps and swings.

## VISUAL EXPRESSION LANGUAGE

CHOPs make a math expression language both visual and interactive. You can build math expressions with CHOPs without having to learn any expression language. Using CHOPs you can interactively build expressions and see the results in a graph or Viewport immediately. There is no typing and therefore no "syntax errors". CHOPs do arithmetic, fetch data from other objects and more.

## MOTION MANAGEMENT FOR LARGE PROJECTS

A team of animators can maintain many motion clips in a central library. Motion can be easily moved between animators and scenes. CHOPs are a convenient way of managing motion of large projects, where clips for many characters are spread across many scenes. Several animators can work on single characters. Through motion compositing, you can build up or "composite" character motion from a library of motion clips, and then hand-modify them. This helps keep a consistent look for character, especially when several artists animate the same character.

# 2 ANATOMY OF CHANNEL OPERATORS

## 2.1 INSIDE A CHOP

- A CHOP contains a set of channels defined over a start-end *interval*.
- The channels of a CHOP can represent *motion* and *audio*, but also can represent *colour maps*, *rolloff curves* or *lookup tables*.
- Each channel is one *array* of *raw samples*, which is simply a list of numbers.
- Each channel of a CHOP has a *channel name* that can be set by the user.
- The group of CHOP channels is known as a *clip*. A CHOP contains a clip plus control parameters, a sample rate, some on/off flags, and a start/end interval.
- The CHOP can have any *start/end indexes*. All channels in a CHOP share the same start-end *interval*.
- The interval goes from the *start position* to the *end position*.
- The *interval* of a CHOP is not restricted to be the length of the animation.
- Each CHOP has a *sample rate*, used if the CHOP contains time-dependent motion or audio data. Audio has a high sample rates when compared to animated motion.

## 2.2 CHOP INPUTS, PARAMETERS AND OUTPUTS

- CHOPs have two sets of channels: the *control channels*, also called CHOP *parameters*, plus the *data channels*, which are input/output from the CHOP as its data stream. The two types of channels are kept separate.
- The *control channels*, or *CHOP parameters* are usually constant-valued channels, but can be keyframed segments like the parameters of any other OP. The *parameters* are in the CHOP's dialog box.
- Each CHOP receives channels at its inputs. When the CHOP *cooks*, the channels of the inputs are combined. The CHOP outputs the resulting channels to other CHOPs.
- CHOPs output their data channels as *arrays of numbers*, not interpolated segments. Some CHOPs retain *interpolated segments* internally, but all CHOPs always output their data as raw samples.
- If the CHOP's inputs are not changing and the control parameters are *not time-dependent*, the CHOP will be non-time-dependent and it will not cook every time the animation frame advances.
- Some CHOPs have *Local Variables*:
  $I (the array index within the CHOP), $C (the channel number within the CHOP)

• Some CHOPs output or use *CHOP attributes*, such as channels grouped as quaternion rotation channels.

## 2.3 CHOP COMPONENTS

• CHOPs are edited in the full-screen *CHOP Editor.*

• The Houdini parameter channels in objects, SOPs, etc. are edited in the full-screen *Animation Editor,* or the floating-window *Channel Editor*, that edits keyframed, interpolated channels.

• Each CHOP has a set of *parameters* (or *control channels*).

• CHOPs' data can be expressed in different *units*: frames, samples or seconds.These units are selected by the user per-CHOP.

• Each CHOP has flags:

- The *Graph flag* marks the CHOP to be displayed in the Graph of the CHOP Editor.
- The *Audio flag* selects the CHOP to be heard.
- The *Export flag* causes the CHOP channels to override channels of objects, SOPs.
- The *Lock flag* causes the CHOP can be locked and hand-edited. The Channel Editor is the interactive editor of parameter channels in CHOPs.
- The *Bypass flag* is a convenient way for a CHOP's effect to be disabled.

• Each CHOP has an *info pop-up*, on the CHOP tile, listing channel names and values, sample rate and interval.

• Each CHOP has a *comment field*.

## 2.4 CHOP NETWORKS

• CHOPs are arranged in CHOP networks, where CHOP outputs are connected to the inputs of other CHOPs.

• Each CHOP network is a separately-named entity.

• The *CHOP folder path* internal to Houdini is: `/ch/`*`chopnet`*`/`*`chopname`*.

## 2.5 IMPORTING / EXPORTING CHOP CHANNELS

### OBJECT, SOP AND COP CHANNELS IMPORTED FROM CHOPS

Channels of SOPs, COPs, objects, etc. are able to get values from CHOP channels with expressions like the following:

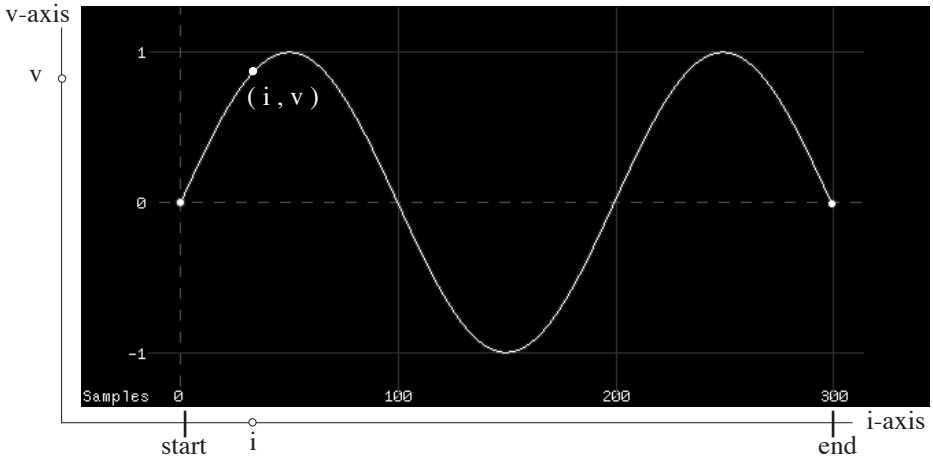
```
chop(chopchannelpath)
chopi(chopchannelpath, index)
```

However, *exporting* is preferred where possible. It is faster and involves less typing.

### CHOPS EXPORTED TO OBJECT, SOP AND COP CHANNELS

- CHOP data channels are *exported* to objects, SOPs, etc. to drive their motion channels. CHOPs can be exported to any Houdini parameter.
- Each CHOP has an Export flag and an Export Prefix, causing the CHOP to attach its channels to objects, SOPs, COPs and so on.
- The Export flag and Export Prefix are used to connect channels of an object or SOP directly to a CHOP. It uses automatic matching of channels names. For example, a CHOP */tx* channel could map to an object's */tx* channel.
- When a CHOP exports to an object, SOP or COP, the latters' channels are said to be *overridden*. An OP's *Override* menu lists what is overridden.

## 2.6 CHANNEL INDEX AND VALUE




- The horizontal axis is called the *i-axis* or the *index-axis*.
- The vertical axis is called the *v-axis* or the *value-axis*.
- An *index* is a point along the *i-axis*, denoted by *i*.
- A *value* is a point along the *v-axis*, denoted by *v*.
- A *sample* is an index-value pair *(i,v)*. i.e. the *value* of a channel at a certain *index*.
- A *sample* is made of a *sample index* and a *sample value*.
- An *interval* is an *index range*, which goes from a *start index* to an *end index*.
- A *value range* goes from a *start value* to an *end value*.
- The *index duration* is the *end index* minus the *start index + 1*.
- CHOP data channels are *arrays* of *raw samples*, in 32-bit floating point format.

- Channels in a CHOP may be *control (parameter) channels* or *data channels*. CHOPs manipulate the data channels.
- CHOPs can be evaluated at integer and non-integer indexes.
- *Frame* is used when the index corresponds to time.
- When speaking of animation frames, you can refer to *start frame*, *end frame* and *frame range*.

## 2.7 CHOPS FOR AUDIO

- Houdini has an *Audio Panel* (*Options* menu) to select which CHOP gets played and how it will play audio: off, play once, play repeat, play when cooked.
- The Audio Panel selects which CHOP gets attached to the timeline. Alternately, a CHOP can be played independent of the timeline.
- The *Audio flag* on each CHOP selects which CHOP gets played.Only one CHOP can be heard at a time.
- On SGI, the Houdini *Audio Panel* feeds its signal to the *SGI Audio Control Panel* found in the *Desktop > Audio Control* menu.

# 3 CHOPS QUICKSTART

## 3.1 INTRODUCTION

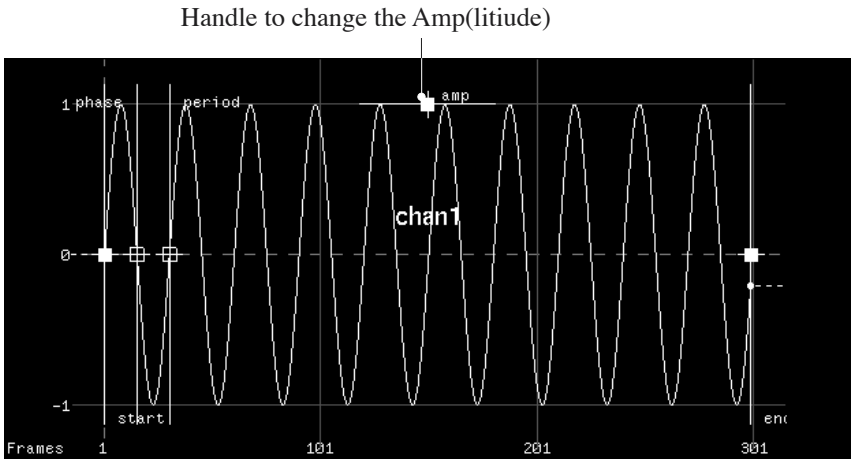
This provides a concise summary of the steps required to get CHOPs to control motion. In the exercise, two sine waveforms are created and are used to animate the motion of the default *logo* object.

## 3.2 EXERCISE

### CREATING THE CHOPS

**1.** Launch Houdini, and open up a CHOP Graph (i.e. *Viewer* ). Enter a CHOP *Network Editor* pane, and place two Wave CHOPs (a generator) into a Network Editor.


**2.** Click the Graph flag of *wave1*. You should see a sine wave in the Graph.




### MANIPULATING THE GRAPH

| | |
|---|---|
| | Scale the Amplitude, Length, and Period of the waveform by dragging with on the handles with those labels. |
| | Resize the Graph. |
| | Scroll the Graph. |
| H | Home the Graph to its original state (cursor must be inside the graph). |

Graph Controls are fully detailed in: *Ref > CHOP Viewer* p. 160.

## EXPORTING CHOP CHANNELS TO OBJECT CHANNELS

To connect a CHOP channel to control a channel of an object, SOP or other OP, you need to "export" the CHOP's channel. To do this, the following needs to be done:

- Name the channels appropriately for exporting.
- Set the *Export Prefix* in the *Common* page. The default *(/obj)*, is often sufficient.
- Set the Export flag on the CHOP tile.
- Examine the Export Destinations in the CHOP tile's info pop-up (🏴 on tile).

**3.** Set the *wave1* Wave CHOP's channel name in the *Channel* page to be: *logo:tx* . In the *Common* page, the *Export Prefix* can be left as: */obj* , which means, "Among the objects *(/obj)*, select the *logo* object, and export to its *tx* channel".


**4.** Click the Export flag on the tile of the *wave1* CHOP, and click *Play*. You should see *geo1* should move left-right as controlled by the sine wave in *wave1*.

For full info on export variations, see *Exporting Channels From CHOPs* p. 260.

## COMBINING CHOPS

**5.** Next, change the *wave2* CHOP to have the channel name: *logo:ty* , and change the *Period* to: 10, and the *Waveform* to *Ramp*.

**6.** Append a Merge CHOP to *wave1* by clicking with 🏴 on *wave1's* Output Connector, and selecting the Merge CHOP.

**7.** Connect *wave2* to the Merge CHOP also. Then turn on the Export flag of *merge1*. You should see the object moving left/right and up/down, illustrating that the motions of both *wave1* and *wave2* have been exported and now control the *tx* and *ty* channels of geo1.

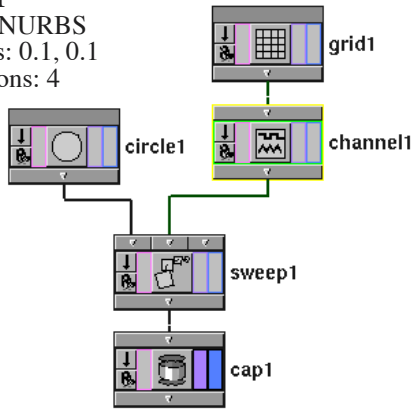
# 4 SEA SNAKES CHOPS DEMO

## 4.1 PROCEDURE

Here is another way to use CHOPs motion channel data to modulate geometry using a Sweep SOP. Do the following:

**1.** Enter the SOPs for a geo object *(geo1)*, and create the following OPnet:

**circle1**
Type: NURBS
Radius: 0.1, 0.1
Divisions: 4




**grid1**
Connectivity: Rows
Size: 4, 1
Rows/Cols: 1, 17

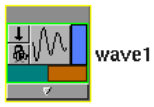
**channel1**
ChopNet/Chop: ch1/wave1
Channel Scope: ty
Attribute Scope: P(1)

**sweep1**
/Output/Skin Output: On

**cap1**
First/Last U Cap: Tangential
First/Last Scale: 0.2
Display flag: enabled

**2.** You will have to bypass the *channel1* SOP until you have created the Wave CHOP (below). Once you've made the Wave CHOP, don't forget to go back and select *ch1/wave1* and un-bypass the channel1 SOP.

**3.** In CHOPs, within the default *ch1*, create a Wave CHOP to drive the motion. Be sure to set the Export (orange) flag!




**wave1**
Type: Gaussian
Phase: $T*0.5
Export Channel Name: ty
Export Prefix: /obj/logo/

**4.** Click *Play*; home and adjust the View as necessary.
You now have your Basic Snake with Motion controlled by CHOPs.

### TRY THIS

Change the motion by setting the *Type* in *wave1* to *Triangle* or some other type.

## 4.2 MORE SNAKES (STAMPED WITH COPY SOP)

By using the Stamp option within the Copy SOP, you can now make many snakes, all with different frequencies of motion and speed, and animating on different paths without having to redo any keyframing. To do this:

**1.** Insert Copy SOP between Sweep and Cap SOP, and set its parameters as follows:

**copy1**
Copies: 3
Translate Z: 3
/Stamp/Stamp Inputs: On
/Var1, Param1: phase, rand(($CY+1.5)*20)
/Var2, Param2: frequency, rand(($CY+1)*0.2)*2 –$CY is the copy number

**2.** In the Wave CHOP, set these parameters:

**wave1**
Period: param(“frequency”,1)
Phase: $T*0.5+param("phase",0) –this references the random# generated by the “phase” parameter of the Copy SOP.

You now have three snakes moving at different frequencies. You can change the number of copies (references $CY in the *Stamp* expression field), and it will generate a random frequency for each additional copy.

--

# 5 WHERE TO FIND MORE CHOPS DEMOS

## 5.1 ALLCHOPS DEMO

To proceed further with CHOPs, there are numerous examples you can study in the *$HD/CHOPs* directory. From there, go to the directory */AllCHOPs* and load the .hip file: *allchops1.hip* .

*allchops1.hip* contains one CHOPnet for each CHOP type, and illustrate the features of each CHOP. Look at all the CHOPnets and play with the parameters.

In the */AllCHOPs* directory, there is an HTML file called *allchops.html*. View it with Netscape while you are running the *allchops1.hip*.

Switch between each CHOP network by selecting them in turn from the Network List (*Options > CHOP Network List* menu) while viewing the descriptions in Netscape.

Enable the Graph's Horizontal and Vertical auto-size options so each CHOP network's channels are fully visible.

You don't have to understand all the details. Just get an overview of some CHOP capabilities. For each CHOP Network, you can check out the parameters of each CHOP, and see how each example was created.

## 5.2 MORE DEMOS

### MOTION CAPTURE

Next you can try a simple one: *$HD/CHOPs/MotionCapture/billflip1.hip*. Play with the *Effect* parameter of both of the Composite CHOPs to get different heights and spins. This can be accessed through the *controls* CHOP.

Also try *arena1.hip* and *billbackhand1.hip*.

### TIMING ADJUSTMENT

Look at *Timing/timing1.hip* and play with the CHOP handles to adjust the timing of the objects.

### SIMPLE DYNAMICS

Look at *Claw/claw1.hip* for a simple lagging technique.

*Ripple/ripple1.hip* lets you shape the object in realtime if you drag around the channels of the Constant CHOP.

### LIP SYNC, MOTION COMPOSITING AND 3D SOUND

Some more of the directories in *$HD/CHOPs/* have HTML web pages you should look through, such as *LipSync*, *MotionComp* and *SpaceAudio*. *SpaceAudio* is the most complex, so you may want to read its HTML page before trying it.

# 6 EXPORTING CHANNELS FROM CHOPS

Once you've created channels in the CHOP Editor, you will want to connect them to the OP parameters (channels in objects, SOPs, etc.) existing throughout Houdini. How this is done follows.

## 6.1 EXPORT AND OVERRIDE

CHOPs alone do not move objects. It is the channels of the objects, SOPs and other OPs that animate everything, such as the *tx*, *ty* and *tz* channels of the object, *geo1*. You need to "export" the CHOP channels to the object channels.

You can "export" CHOPs which automatically "override" the channels of the objects, SOPs, etc.. Export and override are two sides of the same thing.

## 6.2 EXPORTING ONE CHANNEL TO ONE OBJECT'S CHANNEL

To export, you must do the following:

- Name the CHOP channels so they will match the object/SOP/etc. channel names.
- Set the CHOP Export flag on the CHOP tile you want to export.
- Set the Export Prefix in the CHOP dialog's *Common* page.
- Examine the export destinations in the CHOP tile's pop-up info box.

### EXAMPLE 1 – GEO1:TX TO /OBJ

Assuming you have a Geometry object named *geo1* – set a Wave CHOP's channel name to be *geo1:tx* , and in the *Common* page, the Export Prefix can be left as */obj* . It means, "Among the objects ( */obj* ), select *geo1*, and export to its *tx* channel."

Then click the Export flag on the tile of the same CHOP. On the same CHOP's tile, click the info pop-up. You should see the export path to the right of the channel listing. Then play the animation. You should see *geo1* move left-right.

### EXAMPLE 2 – TX TO /OBJ/GEO1

To get the same effect, you could have used a shorter channel name and a longer Export Prefix. To do this, set the channel name of the Wave CHOP to be "tx". Under the *Common* page of a Wave CHOP, type: */obj/geo1* as the Export Prefix.

The first method is better generally because you don't have to change the Export Prefix from its default of */obj* . In a chain of CHOPs that pass *geo1:tx* from one to the next, you only have to click a CHOP's Export flag to see the CHOP's effect at that stage.

## 6.3 IDENTIFYING OVERRIDDEN OP CHANNELS

You can go to the *geo1* object and see its effect. The overridden channels' parameter is colored orange. In every OP dialog box, there is an *Override* menu. Click *Override* on the dialog box of *geo1*. Choose an entry in the Override menu. Houdini will go back to the CHOP that is the source of the override, and it will become the current CHOP.

## 6.4 FORMING EXPORT CHANNEL PATHS

CHOP channel names with : in it, like *geo1:tx* , refer to the *tx* channel of the *geo1* object. If the Export Prefix is */obj* and the exported channel is *geo1:grid1:rows* , the channel that is overridden is in a SOP, */obj/geo1/grid1/tx* .

***Note: Parts of channel paths in Houdini are generally separated by a / . In CHOP channel names, the separator is a : . This is needed to make sure there is no ambiguity in forming channel paths in expressions, scripts and other parts of Houdini.***

## 6.5 EXPORTING SEVERAL CHOPS SIMULTANEOUSLY

You can export more than one CHOP in a CHOP Network, say the *tx* channel from one CHOP, and the *ty* channel of another CHOP. You can have more than one Export flag on in each CHOP network.

## 6.6 EXPORTING ONE CHOP TO MULTIPLE OBJECTS

You can merge the CHOPs together with the Merge CHOP and export that single CHOP. To do this, set one Wave CHOP to output channel *geo1:tx* , the other Wave CHOP to output *geo2:ty* , and pass both of them to a Merge CHOP. Export the Merge CHOP.

### EXPORTING ONE CHANNEL TO MULTIPLE OBJECTS

If the CHOP channel name is *tx* and the Export Prefix is */obj/geo** , the *tx* channel is exported to the *tx* channel of both *geo1* and *geo2* .

Alternately, you can name one channel, *geo1:tx* and pass it to a Rename CHOP to call it *geo2:tx* , merge them with a Merge CHOP, and export the Merge CHOP.

### EXPORTING TO OBJECTS, SOPS AND COPS FROM ONE CHOP

You can export CHOP channels to any SOP, COP, TOP, material or any CHOP's control parameters too. Here is an example of exporting to objects, SOPs and composites (COPs):

- Export Scope: `/`
- CHOP channel names:

```
obj:geo1:ty
obj:geo1:xform1:sx
comp:comp1:xform1:tx
```

### RETRIEVING CHOP VALUES INTO OBJECTS, SOPS, ETC.

You can use the family of *chop()* functions (see section *CHOP Expression Functions* p. 471) in any OP parameter (channel) to fetch CHOP channels, but exporting is easier and runs faster.

### SETTING THE EXPORT FLAG IN A SCRIPT

- The *opset -o* command sets the CHOP Export flag.
- The *opset -a* command sets the CHOP Audio flag.

# 7 HAND-EDITING CHANNELS

You can hand-edit parts of 3D models in the SOP Editor by going into the Modeler. Similarly, you can hand-edit CHOP channels in the CHOP Editor by going into the Channel Editor.

## 7.1 EDITING CHOP CHANNELS WITH THE CHANNEL EDITOR

You can use the same channel editor that you use to edit channels in the rest of Houdini. In this case, you will take the data channels of the output of a CHOP, curve fit them, and then keyframe-edit the resulting channel segments. This locks the CHOP, and the data channels output from the CHOP are now sampled from the channel segments. While the CHOP is locked, the parameters of the CHOP and the inputs to the CHOP have no effect.

Create a Wave CHOP and make two channels by putting "chan1 chan2" in its *Channel* parameter. On the tile of the Wave CHOP, select from the pop-up menu "*Edit Data Channels*". This puts you into the Channel Editor.

When you first enter the Channel Editor through *Edit Data Channels*, the channels are curve-fit. It converts the input channels to "cubic spline-interpolated segments" using a "least-squares" method.

Click the ✣ button to get the *Fit Panel*. The *Tolerance* option changes the density of time marks. The *Scope* selects which channels will be keyframe-interpolated. The *Start* and *End* parameters select a portion of the channel interval to keyframe-interpolate.

At this stage, it locks the CHOP so it does not recook. You cannot change channel names or CHOP length unless you unlock and lose your Channel Editor changes (this will be improved upon in a later version).

You can then edit the curves using keyframes and interpolation. See the *Ref > Animation* section.

### CHOP OUTPUT

CHOP channels that have been modified with the Channel Editor are retained internally as interpolated segments (like the rest of Houdini).

The curves are output from the CHOP as raw samples again, like any other CHOP. The output will contain the original channels, affected only by the edits done on the channel *Scope* over the *Start - End* interval.

## 7.2 HAND-EDITING METHODS

### FROM A SET OF CONSTANT CHANNELS

You can hand-edit any CHOP type. But it is most convenient to originate hand-edited channels by the following method:

1. Create a Constant CHOP. Set the desired length by turning off *Single Frame* on the *Channel* page and setting *End* to the desired length.
2. Create the desired channel names in the *0* page.
3. Go into *Edit Data Channels*. The Constant channels will have two keyframes per channel. Inset keyframes and edit from there.

   ***Note: Don't hand-edit audio – it is very slow.***

### USING THE SPLINE CHOP

The Spline CHOP is sometimes adequeate for editing curves. It is good for fine-tuning single sections of one or more channels.

It only allows for two interpolation types, *cubic()* and *bezier()*, and the number of keyframes is fixed after the curve fitting is accepted. But it's in the same Graph window as other CHOPs, so it may be more convenient.

If you are editing multiple channels, under *Graphs*, select *Graph Per Single Channel*. Now you can edit the curves separately.

## 7.3 CREATING CURVES WITH THE PULSE CHOP

The Pulse CHOP can be used to quickly make simple one-channel curves, like an ease-out, transition or linear segment.

If you set *Pulses* to about 4, *Interpolate* to *Linear*, *Ease* or *Cubic*, and set *Last Pulse at Last Sample*, you move the handles on the graph to edit the curve. This has the disadvantage that the keys are exactly evenly spaced, and it produces only one channel.

# 8 COMPOSITING PRE-KEYFRAMED MOTION

This also has a web page to help you go through the on-line demo, *$HD/CHOPs/MotionComp/motioncomp1.html*. Run Houdini on *motioncomp1.hip* .

## 8.1 THE MOTION COMP DEMO

In the *MotionComp* demo, located in *$HD/CHOPs*, ten character moves were keyframed with parameter channels in Houdini objects in the traditional way. The moves were keyframed one after the other, back to back in one set of object channels. The ten moves were broken into pieces (gestures), one gesture per CHOP, and then recombined.

Put the Viewport in Select and Transform mode, and while Houdini is playing, click on the icons in the lower part of the Viewport. The object's select script causes a CHOP to shift in time, immediately starting the motion associated with that icon.

### "PREPARE" NETWORK

The set of 33 animation channels containing the original moves are brought into CHOPs with the Fetch CHOP, and then they are split into ten gestures using the Trim CHOP. Each gesture is further simplified by deleting the channels that are not involved in the gesture. For example, hand gestures have the leg channels deleted.

In *MotionComp*, the above is done in a CHOP network called *prepare*.

### "MOVES" NETWORK

In the *moves* CHOP network, the moves are re-combined. This is called Motion Compositing. *moves* is designed for the performance of the compositing of the motion clips.

The compositing is done through a chain of Composite CHOPs, each of which outputs its result as the "base" input of the next Composite CHOP. Each Composite CHOP also builds its layer by Fetching one of the prepared gesture and shifting it in time to the appropriate moment where that gesture is needed. The Composite CHOP's parameters on its *Effect* page blend the gesture in and out.

Then CHOPs are attached to the character using the last Composite CHOP's Export flag on its CHOP tile. Click on the tile's info button to see where these channels are exported.

Look at the HTML page for further information.

### OTHER APPROACHES

Motions can also be combined using the Blend CHOP, especially repeating motions that can be blended together.

In another method, you can get the animation channels into a Fetch CHOP and snapshot individual poses (1-sample intervals) into a groups of Constant CHOPs. The snapshots are time-sorted and interpolated using the Interpolate CHOP.

# 9 LIP SYNC

## 9.1 INTRODUCTION

The demo in *$HD/CHOPs/LipSync* is a method to manage a long timeline with mouth poses that are interpolated. Read the web page in: *$HD/CHOPs/LipSync* .

***Tip: You may want to switch to wireframe if it displays too slowly.***

### ADDING A MOUTH POSE TO THE ANIMATION

The available mouth poses are on the left of the camera port. First put the camera port in Select and Transform mode. Position the current frame at a time when you want to insert a pose. Then click the mouth pose icon in the Viewport.

There are also macros on the keyboard's function keys. Under *Edit*, select *Edit Aliases/Variables* to see the commands associated with the function keys:

| | |
|---|---|
| F1 and F2 | Select which objects to view (two mouth patches or the head with the mouth pasted). |
| F3 and F4 | Select which CHOP drives the mouth (an automatic-envelope CHOP or the Interpolate CHOP that interpolates the hand-animated mouth poses). |
| F6 to F12 | Insert one of the seven mouth poses into the Interpolate CHOP. |

You can Shift-click on any black bar on the graph of the Interpolate CHOP, which makes the pose, a Constant CHOP become the current CHOP. Then you can adjust the parameters of the pose in the CHOP dialog, delete the Constant CHOP by pressing Delete, and *Edit -> Undo Delete Node* to restore it. You can also change the time of the pose by clicking on the black bar and dragging it.

The mouth movement can be exaggerated using the parameters of the CHOP named *override* to "add to" the effect of the Interpolate CHOP. (Shift-click on name of the *override* CHOP bar in the graph.)

See the HTML page and Reference Manual for the Interpolate CHOP for more details.

### ANNOTATING TEXT ON THE VOICE TRACK

Text can be added in a CHOP along its timeline, as described in *Putting Text on a CHOP's Timeline* p. 278. You can select a CHOP to text-display, then change its text. The text stays with the CHOP, and does not get passed from one CHOP to another.

This text can be used for lip sync, where the animator can type in the words while scrubbing on an animation track, and then set mouth poses that correspond to the text.

Alternately, notes can be added to an animation track by a director and used by the animator to create effects as specified by the notes.

# 10 USING MOTION CAPTURE DATA

## 10.1 MOTION CAPTURE CONVERTERS

Generally, a motion capture file is read by a converter (e.g. *mcbiovision*) and two files are created:

- A script to create a Houdini hierarchy, either a forward or inverse kinetics hierarchy. generally one of these scripts is used per character, as it is assumed that all motion clips of the character have the same hierarchy and approximate limb lengths.
- Raw channels that drive the hierarchy, one file per motion capture take, imported into one CHOP. The CHOP contains one channel for each joint degree-of-freedom.

Use *mcbiovision* (Biovision) or *mcacclaim* (Acclaim) and *mcmotanal* (Motion Analysis) to create a script and a channel file. Start Houdini, get the textport and type:

```
cmdread xx.cmd
```

where *xx.cmd* is the script that the convert program created. Then create a File CHOP and read the channel file. There are some demos in: *$HD/CHOPs/MotionCapture* and *DynamoJapan* .

The commands will create a separate channel file for each motion clip, read each one into its own File CHOP, and export these File CHOPs. You should be able to drive the character motion directly.

## 10.2 MOTION ANALYSIS

When you convert the Motion Analysis file, the default output format is *-o* (object joints). You can also use *-w* (object joints + wire polys). This will make polygon lines between the joints so you can see the motion more clearly.

The Motion Analysis converter currently only supports `.trc` files. This format does not have parent-child descriptions (for connecting joints). So, in order to output the wire version you must also supply it with a parent file of your own *(-p parent_file)*.

This parent file is a list of pairs of joints of the form: *child-joint parent-joint* that tells the converter which joint connects to which joint.

For all the options of the converter, just type *mcmotanal* with no parameters.

Following, are some production notes on the use of Motion Analysis data with CHOPs.

## PREPERATION FOR MOTION ANALYSIS DATA

When you receive a *.trc* file, you need to convert this file into a *.cmd* file to be sourced into Houdini. This is done with the *mcmotanal* command. This is done in the following steps:

**1.** Since Motion Analysis doesn't provide you with any parent-child descriptions, you need to create a "`parent_file`" for the `mcmotanal` to work properly (see Parent File Example).

**2.** Once the `parent_file` has been created, you can use the `mcmotanal` to create a `.cmd` file that will be sourced into Houdini. There are five basic ways to do this:

- Create an Inverse Kinematics bone hierarchy (script file).
- Create a Forward Kinematics bone hierarchy (script file).
- Create a hierarchy with no bones, but objects instead (script file).
- Create a hierarchy with no bones, but with user-specified objects instead (script file).
- Create a heirarchy with wires (script file).

**3.** You can now start Houdini with the `.cmd` file: `houdini file_name.cmd`. This will automatically source in the generated `.cmd` file into your Houdini file.

**4.** Before you begin you will need to add some frames to the animation. This will allow for the posing of the bone network to match the rest position of the character. You would only really need to add one frame, for example frame zero. Go to the Playbar controls and stretch the animation to 0. Now you can position the bones at frame zero without worry about interfering with the motion data. One way to position your bones would be to initilize the four viewports and maipulate the bone from the viewport using the Handle.

**5.** At this point it would be a good idea to rename all the bones in the network to have a preface of "`bone_`". This will allow easier manipulation from within the CHOP network. See next step ( script file ) .

**6.** The next few steps will be explained, but you can source the opscript file to do this for you. Stretch the animation to start at frame zero and source in the script. First display and then lock the Fetch CHOP before setting the export flag to the Sequence CHOP (script file).

**7.** Once the bones are in position, you will need to go to the CHOP Editor and add a Fetch CHOP. In this CHOP you will fetch you "OP" parameters will be "obj" and "none", and the Fetch parameters will be "`bone_*/r? bone_Root/t?`". You will also need to go to the Channel page and change the Channel Range parameter to "Use Start/end" and set them to be: Start= 0 and End= 0. This will find the rotate values for those bones at frame 0. Also go to the Common page and change the Units parameter to "Frames".

**8.** You will now have to use the Rename CHOP to rename all the channels to have the prefix "`bone_`". For example the From parameter should be "`*`" and the To parameter should be "`bone_*`".

### ACCLAIM

The procedure is similar to Motion Analysis.

### BIOVISION

The procedure is similar to Motion Analysis.

## 10.3 APPLYING MOTION CAPTURE TAKES TO DIFFERENT CHARACTERS

You first set up one "base" character with all the bones and objects you need to drive it. You Export CHOP channels of a default motion to it.

You make sure the names of the base character are standardized. It is best to start with each of its objects ending in "1". The CHOP network that drives the characters should also end in a "1".

You can collapse the objects into an object subnetwork whose name ends in "1".

Then you can Copy and Paste the object subnetwork and the CHOP network. Both names should end in "2" and so on. When referring to indexes in the CHOP network and objects, you can get the index automatically using the *opdigits()* function.

## 10.4 BLENDING MOTION CAPTURE SEQUENCES

Many motion captured characters are structured as a hierarchy of rotation angles and a set of root translations (e.g. Acclaim). This setup is ideal in some situations to blend motion from one CHOP to the next.

Three problems generally occur when blending motion clips with a Sequence CHOP.

### FLIPPING LIMBS

Blending between rotation angles can give non-intuitive results. If you notice wild gyrations in limbs during the transition of one CHOP to the next, this may be the cause. In this case append each CHOP with an Attribute CHOP and append quaternion attributes. Set the rotation order to that of your object hierarchy.

You may need to ensure your rotation channels are properly named in order for the Attritube CHOP to function correctly. Check the CHOP info to see that the quaternion groups are created correctly. Within the Sequence CHOP turn on the 'Quaternion Blend' option in the Rotation tab. The new rotation values should minimize rotations flipping considerably.

### SLIDING POSITIONS

You may notice that the character slides dramatically from the end of one sequence to the beginning of the next. To fix this problem set the Step value to somewhere near 1.0 in the Scope tab of the Sequence CHOP. Ensure the Step Scope is correctly set to the character's root translation channels. This will position the beginning of each subsequent sequence at the end of the previous sequence, minimizing slippage.

### POSITION SPEED

Moving from one sequence to the next may result in unnatural looking slowdowns or speedups at the point of transition. Setting the *Translate Blend* scope parameters of the Sequence CHOP alleviates this problem.

You can also use the Warp CHOP to speed up or slow down CHOPs at their start or end, prior to sequencing them together.

***Note: Other CHOPs take of advatage of quaternion attributes such as the Composite CHOP.***

# 11 LIVE PERFORMANCE OF ANIMATION

## 11.1 INTRODUCTION

You can drive channels directly from input devices with a combination of the Mouse, Keyboard, MIDI In and the Record CHOPs. Live performance is what we call BeatBox in Houdini. You can use this technique to capture motion that is performed, and then edit it later.

You can try the procedure here, and use *$HD/CHOPs/BeatBox* as a reference.

To set it up, in a CHOP network, put down a Mouse CHOP beside a Keyboard CHOP. Connect the output of the Mouse CHOP to the first input of a Record CHOP. Connect the output of the Keyboard CHOP to the second input of the Record CHOP.

You have set up a typical mouse-driven control for an object. The XY position of the mouse is fed to the Record CHOP. The Keyboard is set up so that if you hold the 1 key down on the keyboard while Scroll Lock is on, then the Record CHOP will record the mouse movement.

Because we will want the mouse to control many parameters, and it can only control two at a time, we use the keyboard to select which two channels it is changing.

The idea with Beatbox in CHOPs (as in PRISMS), is that you hold down a key on your computer keyboard or MIDI keyboard to activate channel changes.

By default, the Keyboard CHOP *(when the Scroll Lock key is on)*, will output 1 when the 1 key is pressed, and 0 otherwise. This value is fed to the Active input of the Mouse CHOP.

Next, in the Mouse CHOP, set *Position X* to `geo1:tx` and *Position Y* to `geo1:ty`. Set the Export flag on the Record CHOP *On*. Graph the Record CHOP. Set the Scroll Lock button on the keyboard *On*. Play forward. Hold the 1 key down and move the mouse. The `geo1` object should move while the 1 key is held down.

## 11.2 VARIATIONS

You can see that the Record CHOP builds a channel that spans the whole animation range. To record a short segment while the 1-key is held down, Change Record in the Record CHOP to Auto Range, hold 1 down, move the mouse, and let go. You will see a short CHOP recorded, which you can edit thereafter.

Increase the strength of the mouse movement. In the Record CHOP's Limits page, increase both the X and Y Gain to 5.

Change the key that is used to active this recording by choosing "2" in the second menu of the Keyboard CHOP and pressing 2 to activate another record.

Make the 2 key act as a toggle by selecting Toggle in Keyboard's first menu, pressing 2 to start the record, letting 2 go, move the mouse, press 2 again to stop it. Play back your recorded segment again.

Go back from *Toggle* to *Momentary.*

Now make the mouse act not as a position control as it has been, but as a speed control. In the Mouse CHOP, in the *Type* menu, select *Speed + Hold*. Now press 2 and (still playing forward), move the mouse. You are now controlling the object speed.

If you select *Single Frame* in the *Record* menu in the Record CHOP, and then you record, you will not end up with a graph, but just the current value. This is sometimes all that is needed, and it may cause other CHOPs to run faster.

There is a convenient way to start and stop Houdini playing the animation: the four arrow keys. ↑ = go forward, → = stop and go forward one frame.

## 11.3 EDITING THE RECORDED CHANNELS

Get fancy by following the Mouse CHOP with a Trim CHOP to select a part that you want, then a Filter or Lag CHOP to make the motion smoother.

In the original Mouse CHOP, in Channel, set the Extend Left and Right conditions to *Cycle* and *Cycle*. Then pass it to a Cycle CHOP and change the Overlap Region to 2 (seconds) to make it loopable. Export the Cycle CHOP. You should see a looping cycle.

## 11.4 RE-RECORDING CHANNELS

Once you have recorded a sequence of motion, set the *Record Type* to *Add*. Next set the *Scope* to just the channels you wish to update. Now, when the Active Input is on, only the scoped channels will update. Furthermore changes will be made relative to the beginning values of the scoped channels. This allows you to 'carry on where you left off'.

# 12 CREATING AUDIO WITH CHOPS

## 12.1 INTRODUCTION

Audio is extensively supported in Houdini. CHOPs can generate sounds and can read sounds in from external sources. Audio can be mixed, triggered and filtered.

The Oscillator CHOP is a very flexible oscillator, signal generator and audio sample player. The Trigger CHOP takes any control signal from Houdini, and generates a control envelope with a delay, attack, decay, sustain and release transition. The Math CHOP acts as a mixer and the Parametric EQ CHOP filters audio.

The audio in Houdini is controlled through a flag on the CHOP tile called the "*Audio flag*", and an *Audio Panel*, which controls when to play the audio, volume levels and looping.

## 12.2 QUICK AUDIO START-UP

To quickly get audio going, try the following:

- Display the *Audio Panel* (Alt I or select from the *Options* menu).
- Connect a default Wave CHOP to a default Oscillator CHOP.
- Set the Audio flag on the Oscillator CHOP tile (second from bottom left).
- On the Audio Panel, select Test. You should hear a rising and falling tone.
- Then select Timeline on the Audio Panel.
- Set the animation timeline into *Real Time* and play the animation forward.
- You should hear the tone.
- Click on the animation timeline and scrub the current frame around.
- Connect the Wave CHOP to a new Oscillator CHOP and set the oscillator's *Base Frequency* to 660. Set its Audio flag.
- Play forward and stop.
- Connect the two Oscillators to a Merge CHOP. Set its Audio flag.
- Play forward. You should hear one oscillator in each left and right channels.

## 12.3 LOADING AUDIO

In its simplest form, an audio track can be read into Houdini with a File CHOP, and attached to the animation timeline through the above steps.

## 12.4 THE CHOP'S AUDIO FLAG

Every CHOP network can have 0 or 1 CHOPs' Audio flag set. To hear a CHOP, make sure its sample rate is an audio sample rate (on SGI, 8000 samples/sec or more). Then select the CHOP's Audio flag. The CHOP that is currently playing is displayed on the Audio Panel.

When you set a CHOP's Audio flag, it causes the Audio Panel to select that CHOP to play.

You can turn off the Audio flag on the CHOP tile and the CHOP List, which turns the sound off.

## 12.5 AUDIO PANEL

The Audio Panel in Houdini is selected through the *Options...* menu, and lets you choose which CHOP gets attached to the animation timeline.

See *Ref > Interface > Audio Panel...* Alt I p. 93.

### AUDIO OFF, TIMELINE OR TEST

Audio can be attached to the Timeline. In this mode, when the animation plays, the audio plays.

If the Audio is in Test mode, it will play independent of the audio timeline.

### SELECTING THE CHOP NETWORK AND CHOP TO PLAY

The two menus select which CHOP you want to play. Changing these menus set the CHOP's Audio flag. Similarly, setting the Audio flag on any CHOP sets these two menus.

### SCRUB CONTROLS

Houdini has sound scrubbing controls, which plays the audio slowly around the current frame. Its scrub decay parameter controls the decay of the sound when scrubbing stops. You can scrub backwards and forwards.

When scrubbing the audio, the "Rate" is the audio scrub rate, and defines the amount of time that will be looped when scrubbing. A 45 means 1/45 second.

When stepping or scrubbing the Playbar, the length of the interval is the Decay parameter in the Audio dialog. A 30 means decay in 1/30 of a second.

### HOUDINI AUDIO PANEL TO THE OPERATING SYSTEM AUDIO PANEL

The Audio Control Panel also sets the overall output level to the speakers. The Audio Panel in Houdini passes its audio through the Audio Panel in IRIX on SGI. The SGI Audio Panel is displayed through the IRIX Toolchest -> Desktop -> Control Audio menu. The Houdini audio levels are reduced (multiplied) by the SGI audio panel levels.

### THE AUDIO VU LEVEL METER

The Audio Panel VU meter, when CHOPs are in the range -1 to +1, is at 100% of the acceptable level before the sound amplitude is clipped, and displays as full green. When the meter is red, audio is still audible but the sounds are clipped.

### REPEATING AUDIO ON THE TIMELINE

If you have an audio clip that you want to repeat along a longer timeline, set the *Repeat* flag. When playing in Timeline mode, the audio will repeat, without using extra memory.

## 12.6 OUTPUTTING AUDIO TO A FILE

You can output the audio in any CHOP using the menu on the CHOP tile. Select *Save Data Channels* on the CHOP tile and hit the arrow button to see the file format suffixes that can be used on your output files, such as *.aiff* . You can check the file by playing it with the *sfplay* IRIX command.

## 12.7 DOING THINGS TO AUDIO

| | |
|---|---|
| Trim CHOP | Shorten the audio. |
| Cycle CHOP | Make it repeat. |
| Math CHOP | Increase/decrease volume, mix sounds, stereo to mono, perform functions of a level mixer. |
| Oscillator CHOP | Make audible tones & LFO (low frequency oscillator). |
| Constant CHOP | Provide constant-pitch control for oscillator. |
| Wave CHOP | Add vibrato or LFO. |
| Noise CHOP | Generate white noise. |
| Band EQ, CHOP | Graphic equalizer. |
| Parametric EQ, CHOP | Filter, echo, pitch shift. |
| Audio Filter CHOPs | Low pass, high pass, band pass filter. |
| Shift CHOP | Re-time the sounds, simple echo, delay. |
| Resample CHOP | Change the sample rate of a sound. |
| Stretch CHOP | Make a sound slower or faster, but shifting pitch too. |
| Trigger CHOP | Make volume envelope with delay, attack, decay, sustain, release. |
| Copy CHOP | Use triggers to copy sounds along a timeline. |
| Pulse CHOP | Time pules to produce rhythms as a sequencer. |
| Composite CHOP | Layer one sound over a longer sound. |
| Sequence CHOP | Cross-dissolve one sound to another. |

| | |
|---|---|
| Lag CHOP | Make glissando pitch sweeps. |
| Limit CHOP | Mash it up, clamping, quantizing, sample-hold. |
| Lookup CHOP | Soft limit amplitudes and mash too. |
| Envelope CHOP | Follow the loudness of a sound. |

## 12.8 3D AUDIO

CHOPs can take events in a 3D scene, imported through the Fetch, Geometry or Object CHOP, and be used to trigger sounds, or to control the placement of sounds in 3D space.

An object's, particle's or point's position in 3D space can be used to set the left-right ear separation, or used as reverb controls which add a cue that simulates proximity of the sound.

Audio can be generated for offscreen objects, adding more 3D cues and realism.

The Audio Control Panel controls what you hear when the animation is stopped, which is where much of the audio experimentation takes place. You choose which CHOP will play. If you choose the CHOP whose audio flag is on, this lets you jump quickly to different CHOPs and hear what they sound like.

You can choose to play the CHOP once per button click, when the CHOP cooks, or continuously. Also you can choose to restart the CHOP if a parameter changes and it recooks, or to just keep playing without restarting.

Take a look in: *Objects > The 3D Spatial Audio System* p. 367 for information on setting up Spatial Audio.

## 12.9 AUDIO DEMOS

### AUDIO 3D

Start the `.hip` file in *$HD/CHOPs/Audio/Audio3D*. Select *Audio Panel ...* from the *Options* pulldown menu. Under *Time Line*, select "`ch1`" and "*Audio CHOP*". Then select *Time Line*. Play forward. You should hear some doppler-shifted drum sound. Stop the animation. Click and drag on the animation time bar and hear the audio play back near that frame (you are scrubbing). You can also increase the *Sustain* parameter, which makes the sound fade out slower.

### AUDIO FILE MIX

Quit Houdini and start the .hip in the *Audio/AudioFileMix* directory. Its audio has no connection to the animation time line. In the Audio Panel, select under *Test*, *four_sounds* and *Audio CHOP* from the two menus. Select *On Change Rewind* and hit the Audio Panel's play-forward button.

You can now go to other CHOPs and enable their Audio flags. You can hear each CHOP's audio.

If you look at the sample rates of these CHOPs (by clicking on the "i" info button on a CHOP tile), they are usually 8000 samples per second and higher. The audio hardware usually needs sample rates that are conventional... 8000, 16,000, 22,000 32,000, 44,100 and a few others (check the hardware spec) Playing it at 4000 Hz is bad. If it gets strange sample rates, it will play the audio at bad speeds.

### SPACE AUDIO

This also has a web page to help you go through the on-line demo, *$HD/CHOPs/SpaceAudio/spaceaudio.html*. Run Houdini on *spaceaudio1.hip*.

## 12.10 AUDIOPANEL SCRIPTING COMMAND

The *audiopanel* command sets up the audio panel from the scripting language. Type *audiopanel* – to the shell for details.

# 13 MORE CHOP TECHNIQUES

## 13.1 SETTING THE DEFAULT SAMPLE RATE AND FRAMES PER SECOND

The default sample rate of a CHOP is the same as the animation frame rate. The animation frame rate defaults to 30 frames per second. This can be changed using the *fps* command. To set to the PAL frame rate for your animation, type:

```
fps 25
```

Then all generator CHOPs created *after* this will have a default sample rate of 25 frames per second.

## 13.2 GETTING XYZ VALUES BETWEEN SOPS AND CHOPS

The Geometry CHOP is used to get the XYZ values in a SOP's points into the channels of a CHOP, where they can be edited with CHOPs. All point attributes in a SOP can be obtained, including user-defined point attributes.

The Channel SOP is used to get values in CHOP channels into the points of a SOP. The Channel SOP takes a geometry at its input, and without changing the surface type or connections between vertices of polygons or meshes, it modifies point values only. It operates much more quickly than the Point SOP.

Any point attribute can be replaced by values in CHOP channels.

In both cases, point groups in SOPs can be used to restrict the data being transferred between SOPs and CHOPs.

## 13.3 GETTING RGBA VALUES BETWEEN COPS AND CHOPS

The Image CHOP is used to get the RGBA values in a COP's pixels into the channels of a CHOP, where they can be edited with CHOPs.

The Pixel Expression COP can be used to obtain channel values from CHOPs, but is relatively slow.

## 13.4 PUTTING TEXT ON A CHOP'S TIMELINE

You may want to put notes along the timeline of a CHOP, so you can use it to cue events. Voice for lip sync can be annotated this way.

Pick a CHOP to annotate. Display the blue vertical *Time Bar* in the Graph – you will use it to scrub the audio. Select a CHOP from the menu. Go to a frame. Type text in the text field and type Enter. You will see the text on the top of the Graph.

## 13.5 SPEEDING UP CHOPS USING THE PERFORMANCE MONITOR

Begin by selecting *Performance Monitor...* from the *Options* menu.

You can get any OP cook time and cook count to display in each OP's pop-up info box. OPs can be made to flash while they are cooking. See the *Tile* menu.

*Enable Output* displays a listing of each OP that cooks, in the cooking order, with object and camera port display times.

You can get a a full listing of what cooks and draws over a short time. To avoid the overhead of the performance window itself, you can gather the statistics quickly in memory and print them quickly:

Play forward. In the *Performance Monitor* window, press *Clear*. Turn *Pause* on. Turn *Enable Output* on for a second or less. Turn *Enable Output* off. Turn *Pause* off and you should see the output of the performance monitor.

In the results, draw times do not include cook times. Unaccounted time + all cook times + all draw times = time reported at End Frame.

Click *Single Frame Capture* and *Enable Output* to get a listing of one frame of timing. The Unaccounted Time may be inaccurate in this case, but cook and draw times are accurate.

More information on the Performance Monitor can be found in: *Interface > Performance Monitor...* Alt Y p. 91.

## 13.6 DIAGNOSING CHOPS WITH THE PRINT() FUNCTION

Sometimes it unclear what value a CHOP parameter may have when it is cooked, if it is an expression. You can put the expression in a *print()* function, and it will display its value when it evaluates. The syntax is *print(string, expression)* where *string* is printed in front of the value on the standard output.

## 13.7 CLONING OBJECTS AND CHOP NETWORKS

Sometimes you want one CHOP network to drive one object. You may want to clone the pair dozens of times, but you may want each to behave a bit differently. So you want to get a number in each of the objects and CHOP networks that identifies it.

If you could get the digits in the object name and the CHOP network name, you could use this number to identify the unit. But there is no obvious way of getting that number into CHOP or object parameters. Here is a way:

The expression function, *opdigits()* gets the numbers that are in an OP name. For example, it extracts the *23* from an OP named *leg23left* .

This allows you to make one CHOP network for each of 100 similar characters. For example, in a CHOP, you can get the number of the CHOP network by using these functions:

```
opname("..")  = "ch1"   (a string)
```

```
opdigits("..") = 1   (a set of digits)
```

*opdigits("opname")* returns the numeric value of the concatenation of all the digits in a node's name. It is used when building several similar networks. For example:

```
opdigits("/obj/geo7") = 7

opdigits("..") = 7          (at the SOP level of geo1)
```

In the 100 objects *geo1* to *geo100*, you can use the expression:

```
opdigits(".")
```

You can use this function in the *Random Seed* parameter in the Noise CHOP, which will generate a different noise curve for each CHOP network.

Now you can make one object/CHOPnet pair named with digits at the bottom of your range, say *geo1* and *ch1*, then paste or use a script to make more.

## 13.8 THE USE OF CHOPS IN MANAGING LARGE PROJECTS

You can hold many moves for many characters in one CHOP network and examine any CHOP by connecting it to a character immediately. So it acts as a library and viewer of moves. Also, when several animators work on one project and the goal is to make the character have similar behaviour from scene to scene, animators can pick moves from the library to start with. This is like a library of materials that keep the rendered look the same from scene to scene.

# 14 USING MIDI WITH CHOPS

## 14.1 SGI SETUP

### INSTALL THE DIGITAL MEDIA TOOLS

Install the SGI Digital Media Tools' for MIDI. This includes the Base MIDI software, MIDI Man Pages, and MIDI Synthesizer. You can check if it is already installed by running this UNIX command:

```
versions | grep -i midi
```

In the output, you should at least see this:

```
dmedia_eoe.sw.midi
```

### ENABLE MIDI PORT IN THE SYSTEM MANAGER

Now go into the desktop toolchest and start the System Manager. Find the section that manages serial terminal ports. Pick the port to which you will attach your MIDI device, and select it as a MIDI port. There are usually three choices... serial port, MIDI port and nothing. Picking MIDI will enable the right demons to open and manage the port as a MIDI port. In UNIX, the demons are the good guys.

### ATTACH YOUR MIDI DEVICE TO THE COMPUTER PORT

Next, you will connect a serial cable from the SGI to either (1) a serial-to-MIDI converter box, or (2) a MIDI musical/controller device that accepts a serial cable.

Now attach the MIDI device to the serial port. On SGI, you have to use the RS-232 serial ports, and you will usually have 2 choices – port 1 or 2. Connect the device to the port corresponding to your choice in the *System Manager*.

The O2 and Octane have IBM-style 9-pin serial connectors. You will need a 9-pin serial to MIDI converter that has an IBM serial connector. One possibility is a Key Midiator (see below). Op Code may also make one.

The Indigo2, Indy and older SGIs use the 9-pin DIN connectors as found on Macintosh computers. Op Code makes a 9-pin DIN serial to MIDI converter for this.

Then on the MIDI side of your converter, run your MIDI cable to your MIDI device.

Some MIDI instruments have a 9-pin DIN connector that goes directly to a computer's 9-pin DIN connector. The Roland MCR MIDI slider box has this option, which makes the connection simpler. But if there is no direct-to-computer port on your MIDI device, you will have to use a serial-to-MIDI converter.

### TEST THE INTERFACE

You can try a program from SGI to see if the interface works. *midikeys* can check that MIDI data is getting into your SGI. *synthpanel* is another program that can read MIDI note data and play music from it. Check in */usr/sbin* or the Desktop Tools for others.

If you can't get these programs to run with your MIDI instruments, there is something wrong with your setup above.

### NOTE ABOUT OCTANE PATCHES

On an Octane, make sure that Patch #2151 is *not installed* – it causes bad problems with the kernel and the MIDI interface will fail when Houdini runs "startmidi".

IRIX 6.5 is reputed to have the most accurate MIDI timing (< 1 msec latency).

## 14.2 NT SETUP

Here is a description of how to set up MIDI on an NT system. As you can see, NT does most of the work for you. And anything that the NT doesn't do for you is handled by the sound card installation software. This varies from card to card.

### INSTALL A MIDI DRIVER

If you had a sound card with MIDI support installed in your computer when Window NT was installed, the Windows NT installation will have installed and set up all necessary drivers for using MIDI. If you install a sound card with MIDI support after having installed Windows NT, the sound card should come with disks or a CD-ROM to install the drivers for that sound card. See the documentation for your sound card to determine how to run the installation program for your sound card.

### ATTACH YOUR MIDI DEVICE TO THE COMPUTER PORT

Most sound cards with MIDI support will come with a cable that has two standard MIDI ports (in and out) at one end, and a connector to plug into your sound card at the other end. No converter should be required. Some sound cards may not come with the required cable, or may require a different kind of connector or converter. If this is the case, see your sound card documention for details on connecting a MIDI device to your computer.

### TEST THE INTERFACE

To test the interface, you can use MIDI software that ships with your sound card, or you can try Houdini directly, as described below.

## 14.3 FOR BOTH SGI AND NT

### TYPES OF MIDI DATA

- Note Events – MIDI keyboards output note events, pitch bend events, and aftertouch events. MIDI sequencer devices like the MIDI-enabled descendents of the Roland 808 give note events as well, timed automatically.
- Controller Events – MIDI controller devices, like the Peavey slider box, the JL Cooper slider box and the Roland MCR box each output controller events, which are 0-127 values. Controller events can be used to control anything, and do not generally signify note on – note off events.
- Others event types – see the Midi In CHOP.

### RUN HOUDINI AND START MIDI INPUT

Now start the default Houdini.

Go into the CHOP Editor. Lay down a Midi In CHOP.

In the *MIDI Source* menu, select the serial port that you just set up. It should say *Serial Port 2*, *SoundBlaster Card* or something to that effect.

Now Houdini is ready to receive MIDI events. Turn on the CHOP for display in the graph. Make the graph autoscale horizontally and vertically so you can see all the changes. Play Houdini forward.

### MIDI EVENTS TRANSLATED TO CHOP CHANNELS

If you look at the pop-up info menu on the CHOP tile, you will see which channels are being created. Among the CHOP channels, "`ch1`" means MIDI channel 1, `c1` through `c8` is controller 1 to 8, and `ch1n` is the note events of MIDI channel 1.

By default, Houdini listens to MIDI channel 1, and receives all note events from 0 to 127, and receives all controller events on controller 1 and 2 of MIDI channel 1.

If you are sending data on other channels, put the channel range in the *MIDI Channel* parameter, such as "`1-8`". `1-16` will catch everything.

If you are playing a MIDI controller box, you must know which controller numbers are going to be produced from your box. On the Control page, the Controller Index should be set to that range, or `1-32` for all controllers. Each controller wll create a separate channel.

***Note:*** To find out what MIDI events are coming into your workstation, turn on the option, *Echo Messages to Textport* in the Midi In CHOP. It will output all MIDI messages in the shell or a new window, and you can use this informatioon to choose the right MIDI channels, note range and controller range in the CHOP.

Now you can play your instrument or move the sliders on your slider box. You should see channels changing in the graph.

### NOTE EVENTS ON SEPARATE CHOP CHANNELS

By default, all note events of one MIDI channel are combined in on one CHOP channel. In Note Output on the Note page, you can make separate channels by selecting Separate Channels. But it will produce 127 channels unless you reduce the range in the Note Scope.

### MIDI CHANNELS MERGED TO ONE SET OF CHOP CHANNELS

Also, by default, the events coming in on MIDI Channel 1 are output on a different set of CHOP channels than MIDI Channel 2. If you don't care which MIDI channel the data is coming in on, you can change the Channel Prefix field from "ch" to "" (blank), merging all the MIDI channels selected in the MIDI Channel parameter.

### RECORDING WHILE PLAYING

*Record Method* on the Record page controls how the CHOP channels are formed. By default, it records into channels at the current frame of a CHOP, so you need to be playing the animation for the recording to take place. When Houdini is stopped, values are still recorded, but they overwrite values at the current frame.

If you don't care about recording your values (sounds like declaring your religion), you can set *Record Method* to *Single Frame* or *Current Frame*.

Sometimes you may want to record channels by putting the Midi In CHOP in *Single Frame* or *Current Frame*, and passing that to a Record CHOP, which has more controls over what to do with the MIDI channels. From here, you are ready to jam.

## 14.4 USING CHANNELS FROM MIDI DEVICES

### SCALING, OFFSETING AND NAMING CHANNELS

You may want to first change the values in each of the controller channels from a 0-to-1 range into any other range. This means scaling and offseting the channels.

If it is just one channel that needs to be scaled/offset, or all channels need to be scaled/offset by the same amount, simply use the Range page of a Math CHOP.

Otherwise, a good way of scaling/offsetting a group of channels is to use a Constant CHOP to hold the scale values for all these channels, and another Constant CHOP to hold the offsets (lower limit) for the channels.

Then you can use a Math CHOP to scale (multiply two inputs, the Midi In CHOP and the Constant CHOP containing scales). You can use another Math CHOP to offset (add two inputs, the prior Math CHOP and the Constant CHOP containing offsets).

The MIDI channels evenually get mapped to Houdini channel names. They can be mapped using a Rename CHOP. Better yet, you can name the channels of the Constant CHOPs as the desired channel names. If they are the first input of the Math CHOP, the result will take on the names of the Constant CHOPs' channels.

See the *inputs* CHOPnet of the demo in *$HD/CHOPs/WeirdGuy* for an example of the use of Constant and Math CHOPs to scale, offset and rename channels.

### LAGS OR FILTERS ON MIDI DATA

To generate channels in realtime that are lagging, you need a second or more of the most recent MIDI data. You can use a Trim CHOP with an Absolute Value range of `$T-1` to `$T`, then pass it to a Lag CHOP and adjust the Lag parameters. Use the "Cook to Current Frame" options.

See the *input* CHOPnet of the *WeirdGuy* demo for an example of trimming and lagging channels being recorded.

### SLIDERS CONTROL BLENDS

MIDI slider boxes are often used to mix other pre-animated things in realtime. The MIDI CHOP can be attached to a Blend CHOP and used as the weighting factors. The *headbob* CHOPnet of WeirdGuy does this on its first five channels.

### USING MIDI TO ASSIST KEYFRAME ANIMATION

This is a technique to edit keyframes from normal Houdini channels, such as position and rotation channels of objects. Instead of selecting channels and editing values using the mouse, one channel at a time, you can use an input device like a MIDI slider box or a Puppetworks device to jump between different animation frames and easily edit multiple channel values with the sliders.

For details, see the description in *Editing Keyframes Using Input Devices* in this manual.

### START, STOP, CONTINUE PLAYING IN HOUDINI

If your MIDI device that feeds your workstation is capable of outputting MIDI Start, Stop and Continue messages, Houdini will start at the current start frame, stop at the current frame, and continue from the current frame, respectively.

### TICKS, BEATS AND BARS

If your MIDI device that feeds your worksation is capable of outputting clock ticks and optionally *Bar Messages*, then you can use MIDI to make animations cycle with the music.

The *Timer Ramp Name* parameter of the Midi In CHOP creates a ramp that starts at 0 at the start of a musical beat, and increases to 1 at the end of a musical beat. (It uses MIDI timer pulses.)

The *Bar Ramp Name* parameter of the Midi In CHOP creates a ramp that starts at 0 at the start of a musical bar, and increases to 1 at the end of a musical bar. (It uses a MIDI sysex code defined by the *Bar Message*, followed by a MIDI timer pulse.)

## 14.5 MIDI DEVICE MANUFACTURERS

There are a number of MIDI devices that are suitable for Houdini. Naturally they all have their pros and cons. Cons frequently include the poor resolution of devices... MIDI controllers usually output integers from 0 to 127.

Any MIDI hardware like slider devices are good for setting things up, but because the sliders cannot be forced to a previous state, they become less useful to edit existing slider settings in a CHOP. However the Constant CHOP allows the editing of slider-data using the relative motion of the sliders.

### PEAVEY PC 1600X

Peavey makes a highly-recommended slider box with 16 sliders and 16 buttons. Model: PC1600x, Cost: about $300 US. Check you local music store or the web site: *www.peavey.com/MIDI* .

Sources in North America include a fast mail-order supplier:
Rainbow: 520-325-3376.
In LA, it is carried by :West LA Music: 310-477-1945.

### OTHER SLIDER DEVICES

Roland makes an inexpensive box, the MCR-8, which has 8 sliders, 8 knobs and 16 buttons. It is about $300 US, but Roland may have discontinued its manufacture.

JL Cooper makes a high-quality and higher-priced model of slider box.

### GENERAL TRANSDUCER DEVICES

Infusion Systems make I-Cube, a set of transducers and output devices that are configurable as parts. Check: *www.infusionsystems.com* .

### KEYBOARD DEVICES

Houdini can be used with virtually any MIDI keyboard to receive note events, pitch bend, velocity and aftertouch. Houdini can also be used with MIDI sequencers.

## 14.6 MIDI TO SERIAL CONVERTERS

Most of the MIDI devices will require a MIDI-to-serial converter box. On O2 and Octane and later, the serial ports on SGIs require the 9-pin female IBM PC connectors. Earlier SGI machines require the Mac printer port cables.

For the O2 and Octane, a suitable converter box is the Key Midiator, from Octet Design in the USA. Phone 503-261-2987 or 800-553-MIDI. *www.octetdesign.com* .

The Key Midiator model is MS-124W (W is important). Get the optional power supply. The price is approximately US $169.

The switch settings on the Key MIDIator should be left at its factory defaults.

In this solution, you need an IBM PC Modem cable (25 pin male to 9 pin female).

To run the Key Midiator on port 1, you need to set the SGI driver to be responding to an RS232 protocol at the 38,400 baud rate. Put this in a script that you run each time the SGI boots:

```
startmidi -p rs232 -s 38400 -d /dev/ttyd1 -n "Serial Port 1"
```

# 15 EDITING KEYFRAMES USING INPUT DEVICES

## 15.1 INTRODUCTION

CHOPs combined with a variety of input devices can be used to keyframe-edit a character or other objects. Examples of external input devices include MIDI or PuppetWorks devices.

## 15.2 CONDITIONING LIVE CHANNELS

You can edit CHOPs and keyframed channels using realtime input devices. A typical setup generally consists of the following chain:

- **Mouse CHOP, Midi In CHOP** or **Pipe In CHOP** These CHOPs will output a set of real-time channel values. (In the case of the Pipe In CHOP, the channels will come from a program that feeds the Pipe In CHOP, including over a Network).

- **Delete CHOP**. This can remove unused channels, which helps keep playback rates optimal.

  **Rename CHOP** or **Constant CHOP**. This CHOP renames the remaining channels so they can be exported to the channels of objects, SOPs, COPs, etc..

  The Rename CHOP has a second input to provide the list of channel names. The second input can be a Constant CHOP containing the desired list of names.

- **Math CHOPs**. A Math CHOP can remap channel values to a more suitable range.

  This gives a minimal set of channels with the desired names and values ranges.

  If the channels are properly named, setting the CHOP's Export flag will cause the channels to immediately drive its destination channels.

## 15.3 USING LIVE CHANNELS TO EDIT KEYFRAMES

This is a technique to edit keyframes of Houdini control channels, such as position and rotation channels of objects. Instead of selecting channels and editing values using the mouse, one channel at a time, you can use an input device like a MIDI slider box or a Puppetworks device to jump between different animation frames and easily edit multiple channel values with the sliders.

For example, if you are driving 16 object rotation channels using 16 MIDI sliders, the Midi In CHOP feeds 16 channels into the first input of a Constant CHOP, and the second "Active" input is enabled by another single-channel MIDI button or key, which causes the values in the Constant CHOP to be updated.

See the demo in *$HD/CHOPs/EditKeyframe* for an example of this technique that uses MIDI sliders to edit keyframe values.

- Set up the MIDI channels, and pass them to a Rename CHOP to match the names of the object/SOP channels to edit. For example, rename the channels *tx*, *ty* and *tz*. Then pass the channels to a Constant CHOP and optionally *Snap* the input channels once.

- Select one button (MIDI on/off button or a key on the keyboard) to "enable" the editing, and feed that one channel to the Constant CHOP's second (Active) input. (If you don't want to use a button, you can toggle the "*Live*" button of the Channel Editor instead.)

- Make sure the Constant CHOP is the current CHOP, or turn off the Constant CHOP's *Active Needs Current* flag.

- Go into the Channel Editor and select the channels of the objects/SOPs/etc that you want to edit. In this example, you can create *tx*, *ty* and *tz* channels of *geo1* and select those channels in the Channel Editor. Select the arrow switcher at the bottom, then select from the menus the Constant CHOP to affect the channels. Turn *Live* on.

- Select the channels you want to edit, at any keyframe. Press the Active button that you chose above. Assuming the Channel Editor channels match the channel names of the Constant CHOP, a movement in the input device changes the value at the keyframe in the Channels Editor.

### ADJUSTING A SNAPSHOT POSE

The Constant CHOP allows a second optional input, 'Active'. While this input is non-zero, any changes in the first input are added to the current value of the output.

'Active' means "update the current CHOP with any changes to its inputs".

This allows you to selectively adjust/update the constant values by controling the state of the 'Active' input. Active is often driven by a keyboard or MIDI button to activate the CHOP. (See *Using MIDI to Assist Keyframe Animation* p. 285).

Typically this active input is itself a live channel derived from a Mouse, Midi In or Pipe In CHOP. In some situations this 'mutable' live input is more suitable when in the Channel Editor, as described below.

### HOUDINI CHANNEL EDITOR

Any Houdini spline-based control channel can be edited with the aid of live channels. To do this begin by simply editing the desired control channel(s) normally. Select a live CHOP in the bottom fields of the channel editor (e.g. a Math CHOP).

When the '*Live*' option is selected any changes to the selected CHOP will cause in an identical change to some of the keyframes in the Channel Editor. The keyframes which are affected by the live CHOPs are any keyframes currently selected (highlighted) or at the current frame. This allows you to simply add a keyframe, modify it with the live input, add another keyframe, modify it, etc.

The channels are matched by their names.

## 15.4 TIPS

- Any number of characters can be animated through the same posing device simultaneously by using multiple Rename CHOPs and multiple Active buttons into differnet Constant CHOPs.
- When using "Live Channels" to modify keyframes in the channel editor, you can modify keyframes over a range of frames, or any subset of channels.
- These external inputs can be configured to be seamlessly integrated into Houdini, as any other channel or expression would.

## 15.5 EXAMPLE

**1.** Begin with a default Houdini session. In the CHOP Editor, place a default Mouse CHOP. Graph the CHOP. Notice the two values reflect the current position of the mouse (you may need to disable the automatic Horizontal and Vertical adapt buttons on the graph by pressing the H or V keys).

**2.** Connect the Mouse CHOP to a Constant CHOP. In the *Snap* page, click the 'Snapshot Input' option and turn off *Active Needs Current*. The constant will now hold a frozen instant of the mouse position. You can modify these values by moving the first two sliders of the '0' folder.

**3.** Next place a default Keyboard CHOP. Enable the *Scroll Lock* key on the keyboard so that the playbar turns orange – the Keyboard CHOP will listen to the keyboard while the playbar is orange. Graph the CHOP. Notice the value jumps from zero to one when the 1 key is pressed. Connect this CHOP to the second input of a Constant CHOP.

**4.** Graph the Constant CHOP. Move the mouse while pressing and releasing the 1 key. Note the value of the Constant CHOP is only updated when the key is pressed.

**5.** Move to the Object Editor and invoke the Channel Editor for *geo1* by adding channels to the translate fields. Set the Live CHOP selector to *ch1 constant1* and enable the *Live* button. Add a keyframe to the channels (Alt-click). Notice the time bar moves to the new location. Hold down the 1 key. This will activate the Constant CHOP and modify the new keyframe values accordingly. You may continue this process on new keyframes or by moving the time bar to a previous keyframe.

## 15.6 SNAPSHOT POSES AND INTERPOLATE WITH THE INTERP CHOP

### GRAB (A SNAPSHOT OF) CHANNELS FROM ANOTHER CHOP

You can grab a snapshot of the values and channel names from one of the above CHOPs, the **Mouse, Midi In** or **Pipe In CHOP.**

Create a Constant CHOP, attach a CHOP to its first input and click the *Snapshot Input* button in the Snap page. The Constant CHOP is initialized to the state of its input at the current time.

### INTERPOLATING BETWEEN CONSTANT CHOPS

You can snapshot into a group of Constant CHOPs, one pose per Constant CHOP. If the poses are set to different times, these groups can be output to an Interpolate CHOP to create a smooth transition between the poses.

## 15.7 USING A PUPPETWORKS DEVICE INPUT WITH CHOPS

If you have access to a PuppetWorks device, you may use it directly within Houdini to capture motion sequences and/or setting keyframes in your channel editor.

In the default Houdini installation's */bin* directory, is a program called *puppetworks*. This is a standalone application used to communicate values between the device and Houdini. This is done through an intermediate FIFO file which the application writes to and the PipeIn CHOP reads. FIFO stands for First In, First Out, and is a special file type in UNIX that behaves like a pipe and allows programs to talk to each other. FIFOs are also called "named pipes".

The *puppetworks* application will create the FIFO file automatically. (In SGI UNIX, here is the command that is run internally to create the default FIFO, */tmp/chop.pipe*: `mknod /tmp/chop.pipe p`).

Begin the *puppetworks* program with *-h* as an argument. The usage message will describe the available parameters. Basically you must supply a device name (example: */dev/ttyd2*) and a FIFO file to create. A typical file name is: */tmp/chop.pipe* . The Puppetworks hardware requires a small amount of time to initialize properly. This initialization period should generally be from five to ten seconds.

Not supplying a filename will cause the application to echo all messages to the screen. Start the *puppetworks* program with the appropriate device name and initialization period but no filename:

```
puppetworks -d /dev/ttyd2 -i 5
```

After the five second initialization period, the interface box green LED will begin flashing and a message resembling the following should be reported:

```
Using device /dev/ttyd2 init_time 5 file: <none>
IB code version   20.01
Found 8 joints:
  name: 90000A
  name: 90000B
  name: 90001A
  name: 90001B
  name: 90002A
  name: 90002B
  name: 90003A
  name: 90003B
```

From this point on, everytime a change is detected a message describing the current state of the joints is sent to the screen:

```
  [0] angle=     1.440
  [1] angle=     1.440
  [2] angle=     1.440
  [3] angle=     1.710
  [4] angle=     0.000
  [5] angle=     0.000
  [6] angle=     0.000
  [7] angle=     0.000
```

Restart the *puppetworks* program with the necessary parameters to write to a FIFO file:

```
puppetworks -d /dev/ttyd2 -i 5 /tmp/chop.pipe
```

After the initialization period the program will begin waiting for a PipeIn CHOP to read the values. Startup Houdini. In the CHOP Editor, place down a PipeIn CHOP. The default values should match the parameters used by the application. Display the CHOP. You should now see a single value for each channel reported by the device.

Adjusting the device will immediately cause the PipeIn CHOP to update as well.

Please refer to the section *Using a PuppetWorks Device Input with CHOPs* p. 290 for more information on posing and animating characters.

If you have trouble initializing the device please ensure the device name is correct and then increase the initialization time by a few seconds. If you continue experiencing difficulties please contact your PuppetWorks distributor.

# 2 Channel Operators (CHOPs)

## 1 FAMILIES OF CHOPS

The pop-up help card on each CHOP dialog box provides as much valuable information about a CHOP as possible. In many cases, more details and more verbose descriptions are found only in this reference manual. It is useful to check this manual for possible features that you may not find in the pop-up help.

### 1.1 GENERATOR CHOPS

Generator CHOPs set their own start/end/length interval and sample rate. They normally don't have any inputs from other CHOPs, but they may take data from other parts of Houdini.

Generator CHOPs have local variables that can be used to make each channel different: $C is the channel number, starting at 0. If $C appears in a generator CHOP's parameters, its control channels are evaluated before each channel is cooked.

| | |
|---|---|
| *File CHOP* p. 346 | Read files from disk and make channels. |
| *Constant CHOP* p. 323 | Snapshot or create a set constant channels. |
| *Wave CHOP* p. 468 | Generate repeating wave patterns. |
| *Noise CHOP* p. 393 | Generate non-repeating noise patterns. |
| *Pulse CHOP* p. 425 | Generate pulses at regular intervals. |

### 1.2 EXTERNAL CHOPS

| | |
|---|---|
| *Fetch CHOP* p. 343 | Continually get channels of any object or OP. |
| *Geometry CHOP* p. 352 | Convert SOP points to CHOP channels.Use a Channel SOP to insert CHOP channels back into SOP points. |
| *Object CHOP* p. 398 | Get position of one object relative to another. |
| *Image CHOP* p. 362 | Convert pixels of an image to CHOP channels. |
| *Particle CHOP* p. 410 | Controls motion of Particles. |
| *Export CHOP* p. 337 | Export channels. |

## 1.3 DEVICE CHOPS

The Device (also known as BeatBox) family of CHOPs places realtime values into channels from input devices. They optionally have one or more input devices that output as channels. These update values even when Houdini is stopped.

| | |
|---|---|
| *Mouse CHOP* p. 392 | Output mouse location. |
| *Keyboard CHOP* p. 367 | Output keyboard up/down states. |
| *MIDI In CHOP* p. 382 | Read MIDI data from devices and files. |
| *Midi Out CHOP* p. 390 | Sends MIDI data to a MIDI device. |
| *Pipe In CHOP* p. 415 | Accept realtime channels from user program. |
| *Pipe Out CHOP* p. 420 | Sends realtime channels out to a process. |
| *Audio In CHOP* p. 313 | Accepts channel data from an audio stream. |

## 1.4 TIMING CHOPS

| | |
|---|---|
| *Trim CHOP* p. 463 | Shorten or lengthen a CHOP. |
| *Shift CHOP* p. 440 | Time-shift the interval of a CHOP. |
| *Stretch CHOP* p. 454 | Change the start-end interval and resample a CHOP. |
| *Cycle CHOP* p. 329 | Make channels smoothly repeat. |
| *Delay CHOP* p. 331 | Delays input channels by specified amount. |
| *Resample CHOP* p. 435 | Change Sample Rate and/or Start and End of CHOP. |

## 1.5 SHAPING CHOPS

| | |
|---|---|
| *Limit CHOP* p. 371 | Clamp, loop, cycle and quantize channel values. |
| *Lag CHOP* p. 369 | Lag and overshoot the input channels. |
| *Spring CHOP* p. 452 | Simulate a spring on each input channel. |
| *Filter CHOP* p. 348 | Apply various filter kernels over samples. |
| *Envelope CHOP* p. 335 | Generate amplitude or power curves from channels. |
| *Spline CHOP* p. 450 | Curve-fit and hand-edit channels. |
| *Lookup CHOP* p. 377 | Use input channels to index into lookup table. |
| *Warp CHOP* p. 466 | Use channel to change timing/acceleration. |
| *Extend CHOP* p. 340 | Extends the range of a CHOP channel. |

## 1.6 COMBINING CHOPS

| | |
|---|---|
| *Math CHOP* p. 378 | Combine CHOPs with multiply/add/subtract/divide. |
| *Merge CHOP* p. 381 | Merge channels of one or more CHOPs. |
| *Logic CHOP* p. 373 | Combine CHOPs by applying logic operations. |
| *Blend CHOP* p. 319 | Blend two or more CHOPs using blend channels. |
| *Composite CHOP* p. 321 | Mix the second input over the first input. |
| *Sequence CHOP* p. 437 | Append and overlap two or more input CHOPs. |
| *Interpolate CHOP* p. 365 | Sort inputs by time and interpolate between them. |
| *Switch CHOP* p. 457 | Switches between multiple inputs. |

## 1.7 EVENT CHOPS

| | |
|---|---|
| *Trigger CHOP* p. 460 | Generate envelope at each trigger threshold of input. |
| *Copy CHOP* p. 325 | Copy second input multiple times along first input. |
| *Count CHOP* p. 327 | Counts number of threshold changes. |
| *Hold CHOP* p. 358 | Sample and hold channel values. |

## 1.8 PERFORM CHOPS

| | |
|---|---|
| *Record CHOP* p. 427 | Record values into channels. |
| *Beat CHOP* p. 317 | Generate beats by analyzing a stream pulses. |
| *Gesture CHOP* p. 354 | Converts input from gestures to channel data. |

## 1.9 AUDIO CHOPS

| | |
|---|---|
| *Oscillator CHOP* p. 401 | Generate tones and repeat sound clips. |
| *Pass Filter CHOP* p. 403 | Apply low-pass and high-pass filters. |
| *Parametric EQ CHOP* p. 406 | Filter audio using a variable-range filter. |
| *Band EQ CHOP* p. 315 | Filter audio using fixed-range filters. |
| *Spectrum CHOP* p. 447 | Determines amplitudes is given frequency spectrum. |
| *Voice CHOP* p. 465 | Converts audio voice input to phonemes. |
| *Phoneme CHOP* p. 413 | Translates english into phonetic values. |

## 1.10 MISC CHOPS

| | |
|---|---|
| *Null CHOP* p. 397 | Pass unchanged channels through. |
| *Sub-Network CHOP* p. 456 | Collapse CHOPs into one CHOP. |
| *Expression CHOP* p. 338 | Combine CHOPs using math expressions. |
| *Attribute CHOP* p. 311 | Add attributes to CHOP. |
| *Area CHOP* p. 309 | Find area, integrate or convert speed into position. |
| *Slope CHOP* p. 448 | Find the slope of a channel. |
| *Transform CHOP* p. 458 | Apply a translate, rotate and scale transform. |

# 2 COMMON CHOP PARAMETERS

## 2.1 INTRODUCTION

Parameters of CHOPs that appear in two or more CHOPs are listed and described here. At the end of each CHOP's parameters, is a heading *Standard Options and Local Variables*, which contains hypertext links to the following common parameters.

## 2.2 COMMON PAGE

### SCOPE AND CHANNEL NAME MATCHING OPTIONS

To determine which channels get affected, some CHOPs have a scope string. This option will exist on a Common page of the relevant CHOPs. Patterns can be used in the scope: * (match all), ? (match single character), the same as the *Channel Name Matching Options* described below.

#### channel name matching options

The following kinds of patterns are used to select existing channels in an input CHOP, used in CHOPs like Rename and parameters like the Scope.

| | |
|---|---|
| `chan2` | Matches a single channel name. |
| `chan3 tx ty tz` | Matches four channel names, separated by spaces. |
| `chan*` | Matches each channel that starts with "chan". |
| `*foot*` | Matches each channel that has "foot" in it. |
| `t?` | The ? matches a single character.<br>t? matches two-character channels starting with t. |
| `r[xyz]` | Matches channels rx, ry and rz. |
| `blend[3-7:2]` | Matches number ranges giving blend3, blend5, and blend7. |
| `blend[2-3,5,13]` | Matches channels blend2, blend3, blend5, blend13. |
| `t[xyz]` | `[xyz]`matches three characters, giving channels tx, ty and tz. |

### SAMPLE RATE MATCH OPTIONS

The Sample Rate Match Options handle cases where multiple input CHOPs' sample rates are different. When the CHOP needs to combine inputs with different sample rates, the Sample Rate Match Options offers these choices:

*Resample At First Input's Rate*
: Use rate of first input to resample others.

*Resample At Maximum Rate*
: Resample to the highest sample rate.

*Resample At Minimum Rate*
: Resample to the lowest sample rate.

*Error if Rates Differ*
: Doesn't accept conflicting sample rates.

When Resampling occurs, the curves are interpolated according to the *Interpolation Method Options* p. 304, or "Linear" if the Interpolate Options are not available.

### UNITS OPTIONS

The *Units Options* found in Common pages allows all CHOP intervals to be expressed in Seconds, Frames or Samples. CHOPs that have parameters that describe time, frames or index have a Units menu in its dialog that selects what the units are.

By default the Units of newly created CHOPs are expressed in seconds. But you can change the Units default when new CHOPs are created. To do this, you have to set the UNITS variable. Do this by typing *one* of the following into the Houdini Textport:

```
set UNITS = seconds
set UNITS = frames
set UNITS = samples
```

When changing units in a CHOP, you will also be changing the actual start, end and duration of CHOP intervals. When changing from Seconds to Frames or Samples, change the parameters first and then change the Units. When changing from Frames or Samples to Seconds, change the Units and then the parameter. In this way you won't be stuck with really long intervals.

### TIME SLICE

Time Slicing is a feature which boosts cooking performance and reduces memory usage. Traditionally, CHOPs calculate the channel over its entire frame range. If the channel does need to be evaluated every frame, then cooking the entire range of the channel is unnecessary. It is more efficient to calculate only the fraction of the channel that is needed. This fraction is known as a "Time Slice".

For a full discussion, see *Time Slicing* p. 298.

### UNLOAD

Causes the memory consumed by a CHOP to be released after it is cooked and the data passed to the next CHOP.

### GRAPH COLOR

Every CHOP has this option. Each CHOP gets a default color assigned for display in the Graph port, but you can override the color in the *Common* page under *Graph Color*. There are 36 RGB color combinations in the Palette.

### GRAPH COLOR STEP

When the graph displays the animation curves and a CHOP has two or more channels, this defines the difference in color from one channel to the next, giving a rainbow spectrum of colors.

## 2.3 TIME SLICING

### INTRODUCTION

Time Slicing is a feature which boosts cooking performance and reduces memory usage. Typically, CHOPs calculate the channel over its entire frame range. This is a good method for animation channels which do not need to be recalculated from frame to frame.

However, if the channel does need to be evaluated every frame, due to user input (with a Mouse or Keyboard CHOP) or other unpredictable variables, and we are only interested in the values over the last few frames, then cooking the entire range of the channel is unnecessary. It is more efficient to calculate only the fraction of the channel that is needed. This fraction is known as a "Time Slice".

A Time Slice is simply a very short channel – several frames at most. It only contains the samples between the last cook time and the current cook time. These extremely short channels can be computed much faster than a full animation range channel, and uses up less memory. This gain in speed and memory efficiency is extremely important for audio processing and puppeteering.

Time Slicing also reduces the memory requirements of large Audio CHOP networks, and provides nearly instantaneous results when you tweak audio parameters in a network. Since even a short audio clip can use a megabyte of storage or more, storing only the necessary fraction of the clip reduces the memory requirements significantly, especially in large CHOP networks.

Time Sliced CHOPs do not maintain any more history about the channel than is needed, and they recook every frame. This makes them well suited to realtime applications. Even so, they can be used to "perform" and record an animation, instead of keyframing it.

### THEORY OF OPERATION

Time Sliced CHOPs cook whenever the Playbar is moved forward. The amount that the Playbar jumps ahead is the size of the current Time Slice. The Time Slice is synchronized with the Playbar so that its first sample begins just after the previous Time Slice's last sample, and its last sample is at the current frame. If the Playbar is playing forwards, the Time Slices generated will never overlap, nor will there be any

gaps between them. All Time Sliced CHOPs will have the same Time Slice interval for a given cook.

When the CHOP network needs to cook, the Time Slice interval is computed and a single Time Slice is passed through the network. Since the current frame is always contained within the slice, the slice can be easily exported to any parameter.

## INTERFACE

### chop parameter dialog

The *Time Slice* button is located in the *Common* page of the parameter dialog of all CHOPs. This button is greyed-out for CHOPs that do not have Time Slice capability, and for some CHOPs which always operate in Time Slice mode.

### time slice options

The Time Slice options can be changed by going to *Edit > Preferences > CHOPs*. They can also be changed from the Textport, using the *timeslice* command.

### additions to the audio panel

For audio output, a new page called *Time Slice* has been added to the Audio Panel. Select *Time Slice* audio output mode in the Audio Panel, and select the audio flag of the CHOP to output. This CHOP should contain time sliced channels.

The *Time Slice* page contains a parameter, *Audio Output Delay* which determines the length of time between when the audio data was cooked in the Audio CHOP and the time that the data is sent out to the audio ports (in seconds). This delay provides the realtime audio output with some buffering, so that audio data is not lost if the time between successive cooks is long.

A longer delay gives more protection against gaps and audio popping. This parameter should be adjusted so that it is just big enough to provide adequate buffering, but with the smallest delay. You may have to experiment a little to find an optimal value.

## TIME SLICED CHOPS

Most CHOPs do not need Time Slice capability; they will work on time sliced input in the same manner that they would on normal channel data. Time Sliced CHOPs and non-Time Sliced CHOPs can be mixed within the same CHOP network. The *Time Slice* option is provided for CHOPs that perform operations that read or modify other samples in the channel other than the current sample, or that maintain internal states.

Generally, if you want to create a Time Sliced CHOP network, all CHOPs with Time Slice capability should have *Time Slice* mode enabled.

A complete list of CHOPs with Time Slice capability are listed below. The following CHOPs run in both normal and Time Slice mode, and perform the same functions in both modes.

- Area CHOP
- Copy CHOP
- Count CHOP
- Lag CHOP

- Logic CHOP
- Oscillator CHOP
- Slope CHOP
- Spring CHOP
- Trigger CHOP

The following CHOPs operate only in Time Slice mode. A Record CHOP can be used to record the data over the full animation range.

- Audio In CHOP
- Beat CHOP

The following CHOPs operate in both normal and Time Slice modes, with some differences in operation between the two modes.

- Band EQ CHOP
- Pass Filter CHOP

The filtered audio quality in Time Slice mode is slightly lower than when in normal mode, because audio Time Slices cannot be blended as accurately.

### envelope chop

In Time Slice mode, only the *Exponential Decay* envelope method is used. The *Normalize Power Envelope* is not available, since normalization cannot be done accurately on small segments of data.

### filter chop

When using *Gaussian* or *Box* filtering, the channel is delayed by half the filter size (i.e. a Filter Size of 30 samples will delay the output by 15 samples). To eliminate this delay, use either a *Left Half Gaussian* or a *Left Half Box* filter. Applying a *Sharpen* or *Edge Detect* filter always delays the output by half the filter size. Applying a *Despike* filter will delay the output by the full filter size.

### limit chop

The *Normalize* function does not work in Time Slice mode.

### noise chop

All noise functions work identically in both modes, with the exception of *Harmonic Summation* and *Brownian*. These noise methods cannot be limited to ±1 in Time Slice mode. When the Playbar wraps around to frame 1, the noise functions will continue uninterrupted; they will not cycle back to the same value.

### parametric eq chop

The filtered audio quality in Time Slice mode is slightly lower than normal. Audio time slices cannot be blended as accurately. Pitch Shifting in time slice mode is not supported. Also, pre-echoing cannot be done due to the fact that the future cannot be predicted.

### resample chop

Resample does a simple linear interpolation of the Time Slice in Time Slice mode. Only the sample rate can be changed.

## CHOPS WHICH OCCASION TIME SLICES

The following CHOPs will output Time Slices under certain conditions, but are not considered Time Sliced CHOPs. The *Time Slice* button in the *Common* page is greyed out for these CHOPs.

### midi in chop

This CHOP can output MIDI data into a time slice, or a single frame. Otherwise, it behaves as before. To output only a time slice, set the *Record* menu in the *Record* page to *Current Time Slice*.

### record chop

This CHOP can be used to record the output of a Time Sliced CHOP over the animation range. It can also be used to interpolate a Time Slice from a CHOP that outputs a single frame (like the Mouse or Keyboard CHOPs). The input can be sampled over the entire Time Slice, or only at the current frame using the *Record Input* parameter. The output range can be selected using the *Record Output* parameter.

### trim chop

The Trim CHOP can trim the current time slice out of any input CHOP. This function is selected by changing the *Unit Values* menu to *Current Time Slice*.

## USES FOR TIME SLICING

### realtime puppeteering and control

Time Slicing speeds up the animation response to user input. It provides you with a variety of effects to layer onto the input, such as lag, springs, and filtering. It is otherwise more difficult to achieve these effects on realtime input.

### animation performance

Sometimes an animation is much easier to perform with an input device than it is to keyframe or produce using CHOPs. The input can be recorded with a Record CHOP and locked. This data can then be manipulated with CHOPs, or performed again as needed.

### realtime audio processing

The performance increase of Time Sliced audio processing versus normal audio-processing is quite significant. With Time Slicing, it is possible to process realtime audio input and use the data to control animations. You can also do the opposite – use features of the animation to control audio output. Without Time Slicing, it is difficult to set up a network to do these actions in realtime.

### general audio processing

Time Slicing allows you to clean up or add effects to audio files more efficiently. You can turn all the Time Slice capable CHOPs on and manipulate the audio while playing it, getting realtime results. When you are satisfied with the new audio, the Time Sliced CHOPs are turned off and the final audio clip is generated for the full

range. This allows much more rapid tweaking of parameters to produce a desired effect.

## 2.4 CHANNEL PAGE

### CHANNEL NAME CREATION OPTIONS

The following kinds of text patterns generate multiple channel names:

| | |
|---|---|
| `chan1 chan2 chan3` | Creates channels: chan1, chan2 and chan3. |
| `chan[1-3]` | Creates: chan1, chan2 and chan3. |
| `ch[1-2]n[2-6:2]` | Creates: ch1n2 ch1n4 ch1n6 ch2n2 ch2n4 ch2n6. |
| `r[xyz]` | Creates: rx, ry and rz. |
| `geo[1-2]:t[xy]` | Creates: geo1:tx geo1:ty geo2:tx geo2:ty. |
| `gain band[4-6] off[lr]` | Creates: gain band4 band5 band6 offl offr. |

### START-END OPTIONS

This specifies a start-end interval which is relative to the start-end of the input CHOP, a secondary reference CHOP, or is expressed in absolute numbers.

Start/end can be specified as follows:

#### absolute

The exact new start-end numbers, ignoring the start-end of the input CHOPs. Expressed in Units (see *Units Options* p. 297) in the *Common* page.

#### relative to start/end

Relative to the start/end of the first input CHOP (default).

Some CHOPs have a Reference input CHOP (usually the last input CHOP), which causes the *Relative* start-end parameters to be relative to the Reference input. Start-end becomes a displacement relative to the reference.

Reference inputs are used to make several CHOPs have the same interval.

#### current frame

Selects only one sample, the sample at the current frame.

#### relative to start/end

Behaves in a way that negative values mean back in time, and positive are always ahead in time.

The default is the *Relative* option, with a default start-end of 0, which is the range of the incoming CHOP. The values in these options can be expressed in any Units (see *Units Options* p. 297) in the *Common* page.

### EXTEND OPTIONS

The *Extend* options found in the Extend CHOP are also available in several generator CHOPs. When using the `chop()` function to sample a channel, the index-value may be outside the interval of the CHOP. But a reasonable value is returned. The user is able to control the value of the channel outside the CHOP's interval.

You can see the state of the Extend Conditions in the pop-up info of any CHOP.

#### extend left

The extend condition before the CHOP interval. They are:

| | |
|---|---|
| *Hold* | Hold the first or last value. |
| *Slope* | Continue the slope before the start, or after the end of the channel. |
| *Cycle* | Cycle the channel repeatedly. |
| *Mirror* | Cycle the channel repeatedly, mirroring every other cycle. |
| *Default Value* | Use the constant value specified in the Default Value parameter. |

#### extend right

Extend condition after the interval. Same options as *Extend Left*.

#### default value

The value used for the Default Value extend condition.

## 2.5 OTHER COMMON OPTIONS

### ALIGN OPTIONS

The *Align Options* handle cases where multiple input CHOPs have different start or end times. All channels output from a CHOP share the same start/end interval, so the inputs must be treated with the Align Options:

| | |
|---|---|
| *Extend to Min/Max* | Find the earliest start and latest end, and extend all inputs to that range using the extend conditions. (see *Extend Options* p. 303). |
| *Stretch to Min/Max* | Find the earliest start and latest end, and stretch every channel's start and end to that range. |
| *Shift to Minimum* | Find the earliest start and shift all channels so they all start at that index. All channels are extended to the length of the longest one. |

| | |
|---|---|
| *Shift to Maximum* | Find the latest end and shift all channels so they all end at that index. Extend all channels to the length of the longest one. |
| *Shift to First Interval* | Shift all channels to the start of the first CHOP and sample all inputs using the first input's range. |
| *Trim to First Interval* | Trim all channels to first CHOP's range. |
| *Stretch to First Interval* | Stretch all channels to the first CHOP's range. |
| *Trim to Smallest Interval* | Trim all channels to the smallest start/end interval. The start and end values may not come from the same channel. |
| *Stretch to Smallest Interval* | Stretch all channels to the smallest start/end interval. The start and end values may not come from the same channel. |

### EFFECT RANGE OPTIONS

You may not want to affect the whole interval of a CHOP. You may want the CHOP to have its effect over a limited index range. To allow this, there are four sub-range values (Start, Peak, Release, End). Before the Start and after the End, there is no effect. The effect ramps up (half-cosine) from the Start to the Peak, holds at its maximum value from Peak to the Release, and then ramps down from Release to the End. The *Effect* parameter (0 to 1, default 1) can scale back the overall effect.

### INTERPOLATION METHOD OPTIONS

Determines how to interpolate a CHOP when resampled, as in the Resample CHOP.

| | |
|---|---|
| *No Interpolation* | Takes the nearest input sample. |
| *Linear* | Uses two closest values to determine inbetween values. |
| *Cubic* | Uses the four closest values to determine in-between values. |
| *Pulse Preserve* | If the input has single-frame pulses, create single-frame pulses in output, no matter what the output sample rate is. |

### MATCH BY OPTIONS

When two or more CHOPs need their channels to be matched up, you need to decide how to match them. The matching can be done on the basis of the *Channel Number*, *Channel Name* or *Channel Union*:

| | |
|---|---|
| *Channel Number* | The channel names are ignored and the channels are matched by the order they appear in the CHOP. In general, if the number of channels don't match up, the CHOPs with fewer channels are recycled. |

*Channel Name* The channels names in the first input are used in the output. They are modified only by the channels in the second input that match channels in the first.

*Channel Union* The channels names in the first input + any new names in the second (and more) input are used in the output.

If matching by *Channel Number* and one of the two or more inputs runs out of channels, it recycles the exhausted input. In this way you can have 60 channels of X, Y and Z channels, and three channels of X, Y and Z, and if you add the two inputs, it will recycle the three X, Y, Z channels 20 times.

### REMAINDER OPTIONS

These options affect cases when the output is normally the same interval length as one of the inputs, but due to the CHOP effect, there may be extra samples remaining at the end, such as extra echoes of audio. This parameter determines what to do with what remains at end of the interval.

*Discard Remainder* Output interval = input interval. Discard the remains.

*Make Output Longer* Make the output longer if the effect is unfinished at the end of the input interval.

*Mix Remainder to Beginning* Add remaining samples to the samples at the start. This doesn't make the output longer. It adds the remaining samples at the end to the samples at the beginning. This is good for making audio cyclic.

## 2.6 FILTER CHOPS WITH ANIMATED CHANNEL INPUT

### INTRODUCTION

The Filter Animation Channels input allow audio filter CHOPs to have their parameters animate over the interval of the CHOP. For example, the center frequency of a band-pass filter may sweep up and down every two seconds.

At first glance, one may think that this can be done just by animating the "center" channel of the Parametric EQ CHOP. But this will not give the desired result. What it would do is re-cook the entire audio input of the CHOP at a single center value for each frame that is displayed. That is, the whole sound interval is recooked each frame, using the value of the center channel at that frame.

Instead, like the Composite CHOP, we allow channels in an extra input to animate the filter parameters during each re-cook of the CHOP.

### FILTER ANIMATION CHANNELS

The second input of some audio CHOPs contains optional channels called Filter Animation Channels, that override the filter parameters of the CHOP. They replace the control channels in the CHOP dialog box, and can be used to animate the filter parameters over the interval of the CHOP.

### OVERRIDING A PARAMETER

To override a parameter with an Filter Animation Channel, it must have the exact same name as the parameter's channel. To control the 40Hz band of a Band EQ CHOP, the channel in the second input must be named "b40".

Filter Animation Channels are evaluated at the beginning of every Filter Chunk (see the Pass Filter CHOP) that is processed. As the chunk size decreases, the rate at which your parameters can change increases, but the low frequency sound quality degrades. The first input and the Filter Animation Channels should have the same start-end interval.

## 2.7 LOCAL VARIABLES

There are local variables that are common to many CHOPs. They are:

| | |
|---|---|
| $NC | The total Number of Channels in the CHOP. |
| $C | The Channel Index that is currently being processed. |
| $I | The Index that is currently being processed. |
| $V | The current Value at the index currently being processed. |

# 3 ACOUSTIC CHOP

## 3.1 DESCRIPTION

The Acoustic CHOP allows you to design audio filters and sound materials for the SpatialAudio 3D audio system (see: *Objects > The 3D Spatial Audio System* p. 367).

Three different types of filters are available:

| | |
|---|---|
| *Transmission* | Describes which frequencies are transmitted (values close to 1) and which are absorbed by the material (values close to 0). |
| *Reflection* | Describes which frequencies are reflected off the material. Currently reflections are not supported. |
| *Absorption* | Describes which frequencies are absorbed by the filter (inverse of the transmission filter). |

A sound material for a geometry object or a transmission filter requires either a transmission or absorption filter. A microphone filter requires an absorption filter.

For more information on spatial audio, see also the SpatialAudio CHOP, Microphone object, Geometry object and the Sound object.

When designing the filter, the CHOP graph should be in Frames mode, so that 1 frame on the graph equals 1KHz and frame 0 corresponds to 0Hz (DC). This means that frame 5.5 corresponds to 5.5kHz (5500Hz).

Filters cannot have negative values. If a filter has values of more than one, it will add power to that frequency (turn on 'No Resonance' to avoid this). A value of 0.4 in a filter will reduce that frequency by 60%.

## 3.2 PARAMETERS – ACOUSTIC PAGE

### TRANSMIT SOUND

Creates a Transmission filter if checked.

### REFLECT SOUND

Creates a Reflection filter if checked.

### ABSORPTION

Creates a Absorbtion filter if checked.

#### FREQUENCY RANGE */freqrange1-2*

The frequency range of the filter(s). Typically set to 0 - 22050 (which is half of the 44.1kHz sampling frequency of digital audio from aCD)

#### POWER

Selects a power conservation method.
Applicable when more than 1 filter is being designed.

*No Power Conservation* Power can be reduced or amplified.

*Conserve Total Power* Total power is conserved but the individual frequencies' power is not.

*Conserve Power Per Frequency*
Each frequency has its power conserved.

#### LOCK

If power conservation is on, moving a handle on a filter will adjust the other filters. This parameter allows you to lock one of the filters.

#### NO RESONANCE

Disallows filter values greater than 1.

## 3.3 PARAMETERS – HANDLES PAGE

#### HANDLES */numhandles*

The number of manipulation handles to create.

#### LOGARITHMIC HANDLES

If enabled, the handles are logarithmically spaced, which is more useful than linear spacing (off) for designing audio filters (since human audio perception is not linear).

#### SAMPLES */samples*

The number of samples in each filter.

#### INTERPOLATION

How the samples are interpolated from the handles; either *Nearest Neighbour*, *Linearly* or *Cubically.*

#### RESET ALL WAVEFORMS

Resets all filters to a constant value.

# 4 AREA CHOP

## 4.1 DESCRIPTION

This CHOP calculates the area under a channel's graph, which is the same as calculating the "integral" of a channel, or "integrating" the channel.

It uses the graph of each input channel and calculates the area between the graph and the horizontal 0-value line. It finds the area between a Start and End index, which is by default the entire CHOP range.

The area is calculated by adding the channel values for every sample, starting with the sample at the Start index. Negative values reduce the area. The area is converted to the Units by dividing by samples per Unit. The cumulative values are put in the output channels.

This CHOP is particularly useful for calculating a point's position from its velocity (speed) or acceleration. If the input is a velocity, a First Order integral will return the position. If the input is an acceleration, a Second Order integral will return the position, and a First Order integral will return the velocity.

The first input contains the channels to be integrated.

The second input is used to reset the area to zero. At samples where the second input is zero or less, the area is reset to zero. A Wave CHOP passed into the second input causes the Area to be zero for half a cycle.

The third input is an optional start/end reference. If connected, it will override the parameters in the Range page and integrate the first input's channels between the start and end of the reference input.

## 4.2 PARAMETERS – AREA PAGE

### ORDER

Determines the order of the integral to use. If the input is a velocity, a First Order integral will return the position. If the input is an acceleration, a Second Order integral will return the position, and a First Order integral will return the velocity.

### FIRST CONSTANT

Constant to add to the entire result after integrating once.

### SECOND CONSTANT

Constant to add to the entire result after integrating twice.

### THIRD CONSTANT

Constant to add to the entire result after integrating three times.

## 4.3 PARAMETERS – INTERVAL PAGE

Specifies the interval of the input over which the curves will be integrated.

### UNIT VALUES

Determines whether the start and end parameters listed below are absolute or relative to the channel's start and end.

### START

The start of the range over which to compute the area.

### END

The end of the range over which to compute the area.

### COOK TO CURRENT FRAME

When enabled, this CHOP only cooks (calculates) the channel between the current frame (on the time line) and the last frame it cooked. When off, when this CHOP is cooked, the entire range is cooked.

The Area CHOP can be used to calculate single, double and triple integrals.
It integrates:

$$a * i\hat{}3 + b * i\hat{}2 + c * i + d$$

The *Units* parameter (Common page) affects the output. The output is expressed in value per sample, value per frame or value per second according to *Units*.

## 4.4 STANDARD OPTIONS AND LOCAL VARIABLES

- *Units Options* p. 297
- *Scope and Channel Name Matching Options* p. 296
- *Start-End Options* p. 302
- *Local Variables* p. 306 – $C, $NC

# 5 ATTRIBUTE CHOP

## 5.1 DESCRIPTION

This CHOP adds, removes or updates attributes of the input CHOP. Currently there is only one attribute type, a "quaternion". This attribute type is used to group rotation channel triplets (rx,ry,rz) together.

Rotations sometimes need to be grouped together since interpolations on independent X, Y and Z rotations do not produce smooth results. Rotations often need Quaternion interpolation to rotate through the most direct path.

Operations such as resampling and blending recognize the rotation triplet with the "quaternion" attribute. They blend or resample the rotation channels using "spherical linear interpolation". Ordinary interpolation can produce poor blending results, whereas quaternion blending produces the shortest rotation path between two sets of rotations.

See some of the CHOPs that use the attribute: the *Sequence CHOP* p. 437, *Composite CHOP* p. 321 and *Interpolate CHOP* p. 365. Other CHOPs may quietly use the Quaternion attribute, such as the Object, Stretch and Resample CHOPs.

The Scope is needed to specify the channels that will be grouped.

## 5.2 PARAMETERS – QUATERNION PAGE

### FUNCTION

The function to perform on the attributes:

| | |
|---|---|
| *Pass Through* | Leaves the attributes untouched. |
| *Replace* | Erases previous attributes and replaces them with the new ones. |
| *Append* | Keeps previous attributes and combines them with the new attributes. If an attribute already exists on a channel and append attempts to overwrite it, an error will occur. |
| *Remove* | Removes all scoped channels' attributes. |

### ROTATE ORDER

Sets the rotation order of the rotation triplet.

## 5.3 STANDARD OPTIONS AND LOCAL VARIABLES

### SCOPE

Selects which channels are X,Y and Z rotations. The channels can be typed in explicitly, or using wildcards. If the number of X,Y and Z rotations selected do not match, an error will occur. Example:

```
*r[xyz]
```

## 5.4 LOCAL VARIABLES

- There are no local variables.

# 6 AUDIO IN CHOP

## 6.1 DESCRIPTION

This CHOP receives audio input from the analog audio ports or the digital port. It always outputs time sliced audio data. If you want to record the data, use a *Record CHOP* p. 427.

***Note:*** Receiving audio from an audio CD using a CD-ROM is not supported. Instead, feed the audio through the Audio-in port.

## 6.2 PARAMETERS – AUDIO PAGE

### AUDIO SOURCE

| | |
|---|---|
| *Microphone* | Listens to the analog microphone port. |
| *Line* | Listens to the analog line in port. |
| *Digital* | Listens to the digital port. |

### AUDIO FORMAT

Selects mono, stereo, or four channel mode. Four Channel mode is not available on all systems. Also determines how many channels are created (1, 2 or 4).

### USE GLOBAL SAMPLE RATE

If on, audio is sampled at the system input sample rate. Otherwise, the next parameter allows you to change the system input sample rate. Note that this may disrupt other applications using the audio system. This parameter does not set the actual channel sample rate; that is set by the 'Sample Rate' parameter in the Channel tab. These two sample rates can differ; the audio is resampled to the channel sample rate.

### INPUT SAMPLE RATE *inrate*

The sample rate of the audio input system.

### RECORD QUEUE *queue*

The number of seconds of audio to buffer at the input. Input audio will be delayed by this amount of time. A large buffer allows for slower cooking; if the buffer is too small, gaps and pops will be heard in the audio.

## 6.3 PARAMETERS – LEVELS PAGE

### CHANNEL 1-4 */chan1vol .. /chan4vol*

Each channel can be scaled separately by any factor.

### INPUT ATTENUATION */inputatten*

This parameter controls the audio system hardware's input attenuation, which corresponds to the "record level" of an audio tape recorder. 0 is the lowest recording level, and 1 is the highest recording level. If you notice audio is being clipped at +-1, lower this value.

## 6.4 PARAMETERS – CHANNEL PAGE

### CHANNEL 1-4 NAME

The names of the audio channels.

### SAMPLE RATE */rate*

The sample rate of the audio channels. This may be different from the audio input's sample rate; the input audio will be resampled to the CHOP's sample rate.

### EXTEND LEFT / RIGHT

The extend conditions for the audio channels.

### DEFAULT VALUE */defval*

The default value for the 'Default Value' extend condition.

## 6.5 LOCAL VARIABLES

None.

# 7 BAND EQ CHOP

## 7.1 DESCRIPTION

This CHOP is a 14-band equalizer which filters audio input channels in the same way that a conventional band equalizer uses a bank of sliders to filter fixed-frequency bands of sound.

The CHOP has 14 bands from 10Hz to 82kHz with one parameter per band. The 20Hz band ranges from 14 Hz to 28 Hz (1 octave centred at 20Hz). All these bands can be shifted up or down in frequency with the *Base Shift* parameter. The shape of each band filter can also be specified as a Gaussian (smooth falloff, where the bands overlap) or Box filter (sharp cutoff, where the bands don't overlap).

The first input contains the channels to be filtered.

### FILTER ANIMATION CHANNELS

The second input are the Filter Animation Channels, which allows the filter parameters to be changed over the CHOP's interval. See *Filter CHOPs with Animated Channel Input* p. 305.

## 7.2 PARAMETERS – LOW PAGE (10-320 HZ)

### BASE SHIFT

The number of octaves to shift the bands up (positive values) or down (negative values). Each octave shifted up doubles each of the bands' frequencies.

### FILTER SHAPE

Selects the band filter type to use. *Gaussian* produces a smooth filter, while *Box* produces a very sharp filter.

### 10 HZ – 3200 HZ BANDS

Controls boost/cut at the 10, 20 40, 80, 160, 320 Hz bands.

## 7.3 PARAMETERS – HIGH PAGE (640 - 82KHZ)

### 640 HZ – 82 KHZ BANDS

640Hz, and 1.28, 2.56, 5.12, 10.24, 20.48, 40.96, 81.92 KHz bands.

## 7.4 PARAMETERS – DIGITAL PAGE

See *Digital Filter Overview* p. 404 (Pass Filter CHOP) for a full description of digital filters and the parameters on this page.

### FILTER CHUNK

The size of the chunk that is processed at one time, in samples.

### CHUNK OVERLAP

The portion of the chunk to overlap with the previous one.

### CHUNK DISCARD

The portion of the chunk to discard.

## 7.5 STANDARD OPTIONS AND LOCAL VARIABLES

- *Scope and Channel Name Matching Options* p. 296
- *Sample Rate Match Options* p. 297
- *Local Variables* p. 306 – $C, $NC, $I

# 8 BEAT CHOP

## 8.1 DESCRIPTION

This realtime CHOP is used to manually tap the beat of a piece of music, and automatically generate a repeating ramp or pulse that continues to keep time with the music after the taps stop.

The CHOP computes the length of time (the period) of a beat. It can output five channels: a ramp from 0 to 1, a beat pulse, a beat count, and a constant channel containing the period.

The input must contain two channels, a Listen channel followed by a Tap channel. Often these channels are just two on/off keys on the computer keyboard (from the Keyboard CHOP), or two keys on a MIDI keyboard (from the Midi In CHOP). "On" must be greater than zero, and "off" must be zero or less.

When the "Listen" channel goes on, it means that you desire to change the tempo, and the CHOP is ready to record the user's taps. After Listen goes on, the first tap on the Tap channel changes the output. A time slice or frame is output at the current frame, containing the beat information.

The next taps change the period of the beat, which is the length of the CHOP. The beat period is computed by taking the average time between taps. While the Listen channel is off, the output of the CHOP does not change.

If the Beat channel pulses are longer than one frame, only the time of the first sample matters.

To produce multiples of the tempo (double, half, quarter, etc), use the Tempo Scale menus to select a range of tempos. Tempos from 1/8th to 8x the tapped tempo can be produced.

Hit the Tap channel at any time to synch the ramp to start at the current frame.

## 8.2 PARAMETERS – BEAT PAGE

### LOWER / UPPER TEMPO SCALE

Allows a range of beat channels from 1/8th of the input tempo to 8x the tempo.

### OUTPUT

Beat outputs either a time slice or one frame.

### SYNC BEAT

Resets the ramp on all taps, the first tap, or none.

### GRACE PERIOD

If a tap occurs close to a beat, do not reset the ramp if within the grace period. The units are based on ramp values; 0.1 means a grace period of ramp values above 0.9 and less than 0.1.

## 8.3 PARAMETERS – CHANNEL PAGE

The next four parameters create a channel if a name is supplied.

### TIMER RAMP NAME

This channel is a ramp from 0 to 1. The length of the channel is the length of the beat period.

### TIMER PULSE NAME

This channel outputs a 1-sample pulse at the start of the ramp cycle.

### CYCLE COUNT NAME

This channel is also a ramp from 0 to 1, but it continues increasing the ramp in the extend conditions. It is used to count the number of beats at the current time. Whenever you evaluate the channel, the integer part of the value is the beat count.

### PERIOD NAME

The name of the channel which contains the value of the period. Depending on the *Units* parameter (*Common* page), it will be in Seconds, Frames or Samples.

### PERIOD START NAME

The name of the channel which contains the start sample/frame/time of the current ramp.

### SAMPLE RATE *rate*

The sample rate of the output channels.

## 8.4 STANDARD OPTIONS AND LOCAL VARIABLES

- *Units Options* p. 297
- There are no local variables.

# 9 BLEND CHOP

## 9.1 DESCRIPTION

The Blend CHOP combines two or more CHOPs in input 2, 3 and so on, by using a set of blending channels in input 1. The blending channels cause different strengths of the CHOPs to contribute to the output of the CHOP. It works like the Blend SOP.

Input 1 acts as the control input, which contains the blend weight channels for the rest of the inputs. In it there is one channel for each of the blended CHOPs coming in on input 2, 3 and so on.

The first channel in input 1 is input 2's blend weight, the second channel in input 1 is the input 3's blend weight, and so on. There should be as many blend channels in input 1 as there are inputs, excluding input 1

The interval of the output of the CHOP is the interval of input 1 (the blend channels).

If input 2 onwards are just poses, it's acceptable, as the CHOP blends between poses by using extend conditions.

## 9.2 PARAMETERS

### METHOD

The blend method:

*Proportional* — Each blend source contributes to the result according to its blend weight. If the blend weights do not add up to one, they are scaled so that they do.

*Difference* — In this default behaviour of the Blend CHOP, the input 2 is always the "base". There are blend channels for all the other inputs, and when they are all zero, you get base. If any one blend channel is 1 and the others are zero, then your output is the same as the input that corresponds to that blend channel.

### OMIT FIRST WEIGHT CHANNEL

When using the Differencing method, the weight channel for the base input has no effect, so the channel is omitted if this option is on.

## 9.3 ADVANTAGES OF DIFFERENCE METHOD

Each blend input affects the result without reducing the effect of the others. You can exaggerate beyond each of the inputs by setting their Blend > 1, and you can also use negative values. When all blend channels are 0, you get smooth transitions as any of the blend channels ease out of zero.

## 9.4 STANDARD OPTIONS AND LOCAL VARIABLES

- *Scope and Channel Name Matching Options* p. 296
- *Sample Rate Match Options* p. 297
- There are no local variables.

# 10 COMPOSITE CHOP

## 10.1 DESCRIPTION

This CHOP layers (blends) the channels of one CHOP on the channels of another CHOP. The first input is the base input and the second is the layer input.

Over the interval of the layer, the layer channels are blended with the base channels. The contribution of the layer is eased-in and eased-out according to the *Start*, *Peak*, *Release* and *End* parameters. The base is unaffected outside the interval of the layer.

The *Effect* parameter determines the amount of contribution of the layer.

If *Base Hold* is 0, the layer input will completely replace the base input when the effect is 1. If the Base Hold is 1, the layer will be added to the base.

The interval of the output starts at the minimum of the base and layer. The interval of the output ends at the maximum of the base and layer. The base's extend conditions are used if the layer lies outside the base.

***Note:*** If the third input is supplied, the *Effect* page will be overridden by the third input's first channel, which should contain the effect values over the range of the layer.

## 10.2 PARAMETERS – COMPOSITE PAGE

### BASE HOLD

Determines how much of the base to blend into the output at points where the layer has an effect.

### MATCH BY

Matches channels in the base input with ones in the layer input by either index or name.

### QUATERNION BLEND

Allows rotations with the quaternion attribute set to use spherical interpolation to produce smooth rotation blending (set in the *Attribute CHOP* p. 311).

## 10.3 PARAMETERS – EFFECT PAGE

***Note:*** If the third input is supplied, the *Effect* page will be overridden by the third input's first channel, which should contain the effect values over the range of the layer.

### EFFECT

Sets how much the layer affects the output. If 0, the output is the base.

### UNIT VALUES

Sets the meaning of the next four parameters – either *Absolute* values, *Relative to the Start/End* of the channel, or *Relative to the Current Frame*. The layer and base are never shifted.

### START

The beginning of the composite interval. Effect is zero at the start.

### PEAK

Where the composite operation reaches maximum effect.
This value is held until the release point.

### RELEASE

The point at which the effect begins to fall back towards zero.

### END

The end of the composite operation's effect. The effect reduces to zero again.

## 10.4 STANDARD OPTIONS AND LOCAL VARIABLES

- *Units Options* p. 297

- *Match By Options* p. 304. If *Match By* is *Channel Union* and the Layer has a channel name that does not exist in the Base, the new channel is included in the output and is extended to fill the base's start/end range.

- *Scope and Channel Name Matching Options* p. 296
- *Effect Range Options* p. 304
- *Start-End Options* p. 302
- *Sample Rate Match Options* p. 297
- *Local Variables* p. 306 – $C, $NC

# 11 CONSTANT CHOP

## 11.1 DESCRIPTION

This CHOP allows you to create up to forty new channels. Each channel can be named and assigned a different value. To create a channel, enter a channel name in any left parameter in the 0, 10, 20 or 30 pages, then adjust the value to its right.

The interval is one sample by default (at index 0). An interval range can be optionally set in the *Channel* page.

A simple Constant CHOP with no inputs is the most common use of the CHOP. However, the channels names and values can be set up by connecting any CHOP to its input and clicking the *Snapshot Input* button (*Snap* page). This allows you to get some channels from another CHOP and adjust them with sliders in the Constant CHOP.

The second input can be used to add offsets to the constant values. When the second input (Active) is greater than zero, any change to the first input will be added to the output of the CHOP. This is useful for adjusting Constant CHOP values from external input devices like a MIDI slider box. For example you can connect the mouse or a MIDI slider box to a Mouse or MIDI In CHOP, and you can raise/lower the Constant CHOP values by holding the Active input on, while moving the mouse or sliders.

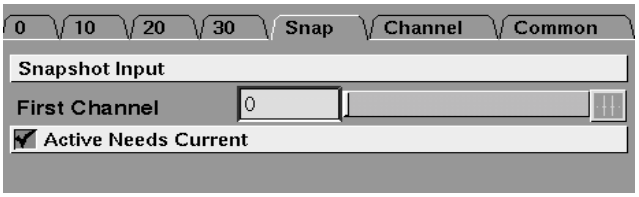
## 11.2 PARAMETERS – 0, 10, 20 AND 30 PAGES

Each page contains fields for ten constant channels. The channel is created if there is a name in the *Name* field. You can enter channel name patterns in the *Name* field, so you can create many channels with the same value. For example, try entering:

```
geo[1-5:2]:s[xyz]
```

in the first channel field instead of "`chan1`". See *Scope and Channel Name Matching Options* p. 296 in the section, Standard Options of CHOPs.

## 11.3 PARAMETERS – SNAP PAGE




The Constant CHOP can preset its channel names and values using the *Snap* page. We often want to grab some channels (Snapshot) and edit them later.

### SNAPSHOT INPUT

The optional first CHOP input on Constant is used when the *Snapshot Input* button is pressed. At this time, the channel names and values at the CHOP input at the current frame are used to initialize the channel names and values of the constant sliders.

To snap channels from other OPs, connect a Fetch CHOP to the Constant CHOP and hit Snapshot Input.

You can simulate the pressing of the *Snapshot Input* button from a script. To simulate the clicking of a CHOP dialog box button from a script use the *opparm* command:

```
opparm -c opname channelname
```

Example:

```
opparm -c /ch/ch1/constant1 snap
```

### FIRST CHANNEL

The First Channel parameter is used to select a smaller set of the incoming channels. This is useful if the number of incoming channels is greater than the 40 channels the Constant CHOP can hold, and you must break it into several CHOPs.

### ACTIVE NEEDS CURRENT

This is used with the second input as described above, when you want to add a displacement to channels by using external devices or sources.

When *Active Needs Current* is on, the second CHOP input (the Active input) has an effect *only* if the Constant CHOP is the current CHOP.

When *Active Needs Current* is off, the Constant CHOP is affected any time the Active input is on (greater than 0).

This is used by the Channel Editor when editing keyframes using CHOPs. An input device like a button in the Keyboard CHOP or a MIDI keyboard can be fed to the Active input of many Constant CHOPs. Only the current CHOP will be affected if this option is on.

## 11.4 PARAMETERS – CHANNEL PAGE

See *Channel Page* p. 302 for a description of these parameters:

- *Start-End Options* p. 302
- *Extend Options* p. 303

## 11.5 STANDARD OPTIONS AND LOCAL VARIABLES

- *Sample Rate Match Options* p. 297
- *Units Options* p. 297
- There are no local variables.

# 12 COPY CHOP

## 12.1 DESCRIPTION

This CHOP produces multiple copies of the second input along the timeline of the first input. The first input provides the trigger signals or the convolve levels.

The Copy CHOP can be used to produce a motion every time a trigger occurs. It can be used to trigger motion, such as eyelid blinks. The copies it produces can be identical, or the copies can be re-cooked each time a copy is added to the timeline. It is useful for triggering a sound multiple times, where the sounds may overlap in time.

Each copy that is added to the output can be completely different than any other copy. By passing variables through the Variables page, the second (Copy) input can be any CHOP chain that uses the variables and recooks to create each copy.

## 12.2 PARAMETERS – COPY PAGE

### COPY METHOD

*Triggered Copy* The second input is copied at the first input's trigger points only. A trigger point occurs whenever the first input's channel crosses the Trigger Threshold value. Overlapping copies are added.

*Convolve* For every sample in the first input's channel, the second input is shifted to that point in time, scaled by the sample value, and added into the output channels.

### OUTPUT METHOD

*One Channel Per Template Channel*
Each output channel is a channel from the first input combined with the corresponding channel from the second input.

*One Channel Per Copy Channel*
Each output channel is a channel from the second input, with copies triggered by every channel of the first input.

### TRIGGER THRESHOLD

The threshold value for triggering copies.

### REMAINDER

See *Common Parameters > Remainder Options* p. 305.

### KEEP NON-SCOPED CHANNELS

If enabled, non-scoped channels are copied to the output, otherwise they are deleted.

## 12.3 PARAMETERS – VARIABLES PAGE

### COOK EACH COPY

Recook the second input for each triggered copy.

### PARAM 1 - 10

The parameters are re-calculated for each copy. The first field is the parameter name, the second is its value. You can use Local Variables and the *ic()* expression functions. The parameters you set here are available to any CHOP in the network attached to the second input through the function:

```
param("name", initval)
```

Where *initval* is any initial value for the parameter, and is usually set to 0.

## 12.4 STANDARD OPTIONS

- *Scope and Channel Name Matching Options* p. 296
- *Sample Rate Match Options* p. 297
- *Remainder Options* p. 305

## 12.5 LOCAL VARIABLES

| | |
|---|---|
| I | The current index. |
| C | The current channel (0 to NC-1). |
| NC | The total number of channels. |
| V | The value at the current index of the current channel. |
| CN | The copy number.<br>(The first thing it copies is numbered 0.) |

# 13 COUNT CHOP

## 13.1 DESCRIPTION

This CHOP counts the number of times a channel crosses a trigger or release threshold. It operates in either static or realtime ("Cook to Current Frame") mode.

Crossing the trigger threshold (increasing past the trigger level) creates a trigger event. Similarily, crossing the release threshold (decreasing past the release level) creates a release event. Operations may also be performed while the input remain above or below the trigger or release levels. On each event, the count may be increased or decreased by 1 or the time, or reset to zero. The time per sample varies with the sample rate (i.e. for 100 samples/second, the time for each sample would be 1/100th of a second).

The optional second input is a reset input. The first channel is interpreted as a channel containing reset pulses. Whenever this channel is non-zero, the count for all channels is reset.

This CHOP can be time sliced, and take advantage of Minimal Time Slice Cooks.

## 13.2 PARAMETERS – TRIGGER PAGE

### RELEASE = TRIGGER THRESHOLD

If on, the trigger threshold is also used as the release threshold.

### TRIGGER THRESHOLD *threshup*

The channel level that must be exceeded in order to trigger a count.

### RELEASE THRESHOLD *threshdown*

A release count is triggered when the channel level drops below this threshold.

### RE-TRIGGER DELAY *retrigger*

The amount of time after a trigger point that a new trigger may occur.

### TRIGGER ON

Determines whether a trigger occurs on an increasing slope or decreasing slope when passing the trigger threshold. A release will occur on the opposite slope.

## 13.3 PARAMETERS – COUNT PAGE

### LIMIT

| | |
|---|---|
| Loop Min/Max | Will cycle in a loop between the values given by Limit Minimum / Maximum. |
| Clamp Min/Max | Clamp will hold the count value at the maximum/minimum value if it goes above or below the limits. |
| Loop Min, Clamp Max | Will loop the count back between the limits by shifting the count to the maximum limit. |
| Clamp Min, Loop Max | Will loop the count back between the limits by shifting the count to the minimum limit. |

**limit minimum / maximum** */limitmin /limitmax*

The minimum and maximum allowed count number.

**LOOP** */loop*

Loop from 0 to loop-1.

### OFF TO ON

The operation to perform when a trigger event (off to on) occurs.

### WHILE ON

The operation to perform while the input remains triggered (on).

### ON TO OFF

The operation to perform when a release event (on to off) occurs.

### WHILE ON

The operation to perform while the input is not triggered (off).

### RESET COUNT TO ZERO

In Cook To Current Frame mode, this resets the count for all channels.

## 13.4 LOCAL VALIABLES

*none*.

# 14 CYCLE CHOP

## 14.1 DESCRIPTION

This CHOP creates cycles. It can repeat the channels any number of times before and after the original. It can also make a single cycle have a smooth transition from its end to its beginning, so it loops smoothly.

Since channels may not naturally loop well, the Cycle CHOP provides three different methods of blending between the cycles.

## 14.2 PARAMETERS – CYCLE PAGE

### CYCLES BEFORE

The number of cycles to loop before the input CHOP. This parameter can be fractional.

### CYCLES AFTER

The number of cycles to loop after the input CHOP. This parameter can be fractional.

### MIRROR CYCLES

If enabled, consecutive cycles are mirror images (reversed) of each another. The first cycle is never mirrored.

### BLEND START TO END

If on, the end of the CHOP is blended into the start of the CHOP to produce a smooth loop. If Cycles Before and Cycles After are 0, Region is non-zero, and Extend Conditions are “Cycle”, it loops smoothly forever.

## 14.3 PARAMETERS – BLEND PAGE

### METHOD

How to blend between cycles:

| | |
|---|---|
| *Preserve Length* | Keeps the total length of each cycle the same as the length of the input CHOP. |
| *Overlap Sequences* | Overlaps each cycle with the previous cycle. |
| *Insert Blend Region* | Inserts a region between the cycles where blending is done. |

### SHAPE

The shape of the blending function:

| | |
|---|---|
| *Linear* | Straight linear blend. |
| *Ease in* | Uses an *easein()* function for blending. |
| *Ease out* | Uses an *easeout()* function for blending. |
| *Ease in Ease out* | Uses both *easein()* and *easeout()* functions. |
| *Cubic* | For *Insert Blend Region*, uses a *cubic()* interpolation to fill the region between the cycles. |
| *Add* | The overlapped regions have the overlapping samples simply added. This is suitable for looping audio. |

### REGION

The size of the blend region, in either seconds, samples or frames (set with *Units* in the *Common* page).

### BIAS

The bias of the blend. A -1 biases the blend toward the beginning of the blend region, 0 is no bias and +1 biases towards the end of the blend region.

### STEP

If set to 1, the next cycle will be shifted up or down in value, so that it begins where the last cycle ended. Suitable for the root object of walk cycles.

### STEP SCOPE

The names of those channels that will be affected by the *Step* parameter.

## 14.4 STANDARD OPTIONS AND LOCAL VARIABLES

- *Units Options* p. 297
- There are no Local Variables.

# 15 DELAY CHOP

## 15.1 DESCRIPTION

This CHOP delays the input, and can be run in normal or time-sliced mode. Like an echo, it delays the input and applies a gain (multiplier). The input may be copied up to four times, each copy with its own delay and gain. The copies are layered onto the output track by adding them together. The Delay CHOP can be used to create echoes, with a bit more control over the delay time and gain than the Parametric EQ CHOP.

It also has the Remainder option of the Copy CHOP when in normal mode, which determines with extending the out put of the CHOP or not.

This CHOP may be time-sliced by enabling the Time Slice flag in the *Common* page.

## 15.2 PARAMETERS

### NUMBER OF COPIES *(numcopies)*

The number of times the input channel is copied. Each copy has its own delay and gain, and the output channel is the sum of these copies.

### REMAINDER

What to do with samples that are delayed beyond the end of the input clip's interval.

*Discard Remainder* Ignore; keep the output clip the same length as the input.

*Make Output Longer* Extend the output channels to fit all delayed samples.

*Mix Remainder to Beginning* Add the remaining samples to the samples at the start of the channel.

### DELAY 1-4 *(delay1 - 4)*

The delay of each copy. This is expressed in *Units*, as set on the *Common* page.

### GAIN 1-4 *(gain1 - 4)*

The gain, or scale of each copy.

## 15.3 LOCAL VARIABLES

None.

# 16 DELETE CHOP

## 16.1 DESCRIPTION

The Delete CHOP removes channels coming from its input. The channels are selected by a variety of methods, which may be layered. The first method uses a text string to select channels by name or index. The second method uses a value range to select channels with samples within or outside the range. The third method selects constant-valued channels.

An option selects whether to delete or output the selected channels.

## 16.2 PARAMETERS – DELETE PAGE

### DELETE

Determines whether the scoped channels should be deleted or retained:

*Scoped Channels* The scoped channels are deleted. The rest are output.

*Non-Scoped Channels* The scoped channels are output. The rest are deleted.

### SELECT CHANNELS

How to select channels – *By Name*, or *By Numeric* index.

### CHANNEL NAMES

Enter a scope pattern here to specify the names of channels to delete or extract. You do this by specifying a scope pattern, as detailed in: *Scope and Channel Name Matching Options* p. 296.

The default scope pattern *t** will scope the translation channels: *tx*, *ty*, and *tz* , or any other channel whose name starts with *t* .

### CHANNEL NUMBERS

The indicies of the channels to delete or extract.
See possible number patterns below.

### CHANNEL VALUE

Chooses the type of value range selection:

*Off* Don't perform range selection.

*Channel Completely Within Range*
All the channel's samples must be within the specified range for it to be selected.

*Channel Partially Within Range*
At least one of the channel's samples must be in the range for it to be selected.

*Channel Completely Outside Range*
None of the channel's samples can be in the range for it to be selected.

### VALUE RANGE *delrange*

The lower and upper values of the range used for Range Selection.

### SELECT CONSTANT VALUED CHANNELS

Select channels which have the same value for all samples. These kinds of channel name patterns are used to select existing channels in an input CHOP:

`chan2` Matches a single channel name.

`chan3 tx ty tz` Matches several channel names, separated by spaces.

`chan*` Matches each channel that starts with "chan".

`*foot*` Matches each channel that has "foot" in it.

`t?` ? Matches a single character.

`t?` Matches 2-character channels starting with t.

`t[xyz]` [xyz] Matches 3 characters, giving tx, ty and tz.

`blend[3-7:2]` Matches number ranges: blend3, blend5, and blend7.

`blend[2-3,5,13]` Matches channels blend2, blend3, blend5, blend13.

These kinds of channel number patterns can be used to select existing channels in an input CHOP:

`0 1 4` Matches the first, second and fifth channel.

`[0-4]` Matches the first five channels.

`[3-9:2]` Matches channels at indices 3,5, 7 and 9.

## 16.3 LOCAL VARIABLES

| | |
|---|---|
| N | The number of points or primitives, depending on the "Entity" parameter. |

# 17 ENVELOPE CHOP

## 17.1 DESCRIPTION

The Envelope CHOP outputs the maximum amplitude in the vicinity of each sample of the input. It takes the absolute value of the input, and uses a sliding window of a number of samples to find the maximum amplitude near each sample.

***Tip:*** The loudness levels of an audio track can be kept roughly constant by computing an envelope of the audio with a wide window, and then passing the original audio and the envelope to a Math CHOP and selecting Combine CHOPs – Divide. This will make the amplitude approximately 1.

## 17.2 PARAMETERS

### TYPE

The two methods of calculating the envelope:

*Exponential Decay* — For each sample, the value is compared to the previous sample. If it is greater than the previous, the value of the envelope is equal to the value of that sample, and that sample is stored as the current peak. If it is less than, the value of the envelope decays exponentially from the last peak to the current value (as more samples pass that are smaller than the peak, the envelope decays toward the waveform).
*Pros:* Always encloses the data.
*Cons:* Slope can be discontinuous, making the output look bumpy.

*Local Maximum Window* — The channel is separated into windows of N samples determined by the *Envelope Width*. In each window, the maximum amplitude (or power) is found. The maximum value of the window is placed as a data point in the center of the window, and these points are cubically interpolated together to form the output envelope.
*Pros:* Produces Good shapes.
*Cons:* Signal sometimes jumps outside the envelope. The signal is quantized, so pulses can be off by as much as N/2.

### ENVELOPE WIDTH

The width of the window to use in the envelope calculation. Adjust this width to capture as many features of the input as needed. It is expressed in Units.

### POWER ENVELOPE

Computes the power envelope of the signal. The *power,* here refers to the square of the *amplitude* – often used to measure the energy of a signal. This option squares the amplitude first.

### NORMALIZE POWER ENVELOPE

Keeps the total power in the signal constant when adjusting the *Envelope Width*.

## 17.3 STANDARD OPTIONS AND LOCAL VARIABLES

- *Scope and Channel Name Matching Options* p. 296
- *Units Options* p. 297
- *Local Variables* p. 306 - $C, $NC

# 18 EXPORT CHOP

## 18.1 DESCRIPTION

The Export CHOP is a convenient tool for exporting channels. It allows you to match a CHOP's channels with different destination channels, without needing to rename the channels. It also provides an enable input to turn exporting on or off on a frame by frame basis.

The first input contains the channels to export.

The second input should contain one channel. This channel controls the exporting of the channels in the first input; if it is 1 at a frame, the channels will be exported for that frame, if it is 0, they will not be exported.

## 18.2 PARAMETERS

### CHANNELS

The list of source channels. Not all channels need to be listed (unlisted channels will not be exported). The order these channels are listed is important; they will be matched to the paths in the 'Path' parameter in the same order.

### OP

These menus allow you to optionally specify the beginning of the path. You can specify the full path (/editor/net/op) or a portion of the path (/editor). If you specify only a portion, each of the channels in the 'Path' parameter will need the rest of the path attached (ie, /editor would need path channels of the form net/op/channel).

### PATH

The list of destination channels. If there are more destination channels than source channels, export will loop back through the source channels until all destination channels have been matched.

# 19 EXPRESSION CHOP

## 19.1 DESCRIPTION

This CHOP allows you to modify input channels by using math expressions. Up to six expressions are available. Each input channel is modified by exactly one expression, and the expressions are looped for multiple channels.

The output is the same length and set of channels as the first input, but its sample values are changed according to the expressions.

An expression is applied to each keyframe value or raw sample. There are up to six expressions. If there are more channels coming from input 0, the expressions are recycled. A repeat parameter controls how many channels to apply the first expression to before going on to the second expression.

## 19.2 PARAMETERS – GROUP PAGE

### CHANNELS PER EXPR

The number of channels that use the current expression before the next expression is selected.

### NUM EXPRESSIONS

The total number of expressions that are defined.

## 19.3 PARAMETERS – EXPR PAGE

### EXPRESSION 1-6

Enter your expressions here.

## 19.4 USING EXPRESSION FUNCTIONS TO ACCESS OTHER INPUT CHOPS

The power of this CHOP extends when using special expression functions to access the other input CHOPs.

*ic(input#, channel#, index)*

The above expression function gets the value at the specified index of the specified channel of the specified input CHOP (all 0-relative).

For more details on CHOP-related expression functions, see *CHOP Expression Functions* p. 471.

Others functions include:

- *ic()* can use local variables: $C (channel #), and $I (index).
- *oc(chan_index, samp_index)* gets at the output CHOP's samples as they are being built. Like in calculus integration, you need a value from the previous output sample to generate the next one.
- *ics(input#)* gets the start index of any input CHOP. similarly for *ice()* and *icl()* .
- *icr()* gives the sample rate.

For further information, see *Channel Operators (CHOPs)* p. 292.

## 19.5 STANDARD OPTIONS AND LOCAL VARIABLES

- *Scope and Channel Name Matching Options* p. 296
- *Sample Rate Match Options* p. 297
- *Local Variables* – $C, $NC, $I, $V

# 20 EXTEND CHOP

## 20.1 DESCRIPTION

This CHOP only sets the "extend conditions" of a CHOP, which determines what values you get when sampling the CHOP before or after its interval.

The Extend CHOP has separate menus to control both the pre-interval and the post-interval. For example, before the interval, you may want to hold a constant value, and after the interval you may want to cycle or repeat the curves in the interval.

To shorten or lengthen the interval (the start or end of the CHOP), use the Trim CHOP.

You can see the state of *Extend Conditions* in the pop-up info of any CHOP (use middle-mouse click on the CHOP tile).

## 20.2 PARAMETERS

These parameters, are the same as those found in: *Extend Options* p. 303 in the *Standard Options for CHOPs* section. A copy follows below:

### extend left

The extend condition before the CHOP interval. They are:

| | |
|---|---|
| *Hold* | Hold the first or last value. |
| *Slope* | Continue the slope before the start, or after the end of the channel. |
| *Cycle* | Cycle the channel repeatedly. |
| *Mirror* | Cycle the channel repeatedly, mirroring every other cycle. |
| *Default Value* | Use the constant value specified in the Default Value parameter. |

### extend right

Extend condition after the interval. Same options as *Extend Left*.

### default value

The value used for the Default Value extend condition.

## 20.3 STANDARD OPTIONS AND LOCAL VARIABLES

- *Extend Options* p. 303
- *Scope and Channel Name Matching Options* p. 296
- There are no local variables.

# 21 FAN CHOP

## 21.1 DESCRIPTION

This CHOP is used for controlling other CHOPs. Its first operation, “Fan Out”, selects one of N output channels based on the input channel’s value. A selected output has a value of 1, and all non-selected outputs have a value of 0. The selection is done by index; the first output channel is index value 0, the second, 1, and so on. If the input value is above N-1 or below 0, the value can be clamped, cycled or ignored.

For example, if the value of the input channel at a certain frame is 4 and the CHOP outputs 8 channels, channel 5 will have a value of 1, and all other channels will have a zero value at that frame.

The input is assumed to have 1 channel which contains integer values; fractions are truncated and extra channels are ignored. The output channels are binary (0 or 1) channels.

The second operation, Fan In, does the opposite: it takes a bunch of binary inputs and produces one channel containing the index of the “on” channel. If more than one input channel is “on”, the first “on” input channel is selected.

## 21.2 PARAMETERS

| | |
|---|---|
| *Operation* | Selects either "Fan In" or "Fan Out". |
| *Channel Names* | The names for the output channels that this CHOP creates. This also controls how many output channels are created (one for each name) in Fan Out mode. In Fan In mode, only one channel is created, and its name is the base name (minus the number suffix) of the first input channel. |
| *Outside Range* | Determines how to handle input values that are outside the index range (0 to N-1). |
| *Clamp Index* | If less than 0, clamp to 0, and if greater than N-1, clamp to N-1. |
| *Loop Index* | Loop back through the index list. |
| *Set Channels to 0* | Don’t select any channels; set every channel to zero. |
| *All Channels Off* | For a Fan In operation, when all input channels are off, set the output to -1 or 0. |
| *Set to 0* | Selects the first channel. |
| *Set to -1* | Doesn’t select any of the channels. |

# 22 FEEDBACK CHOP

## 22.1 DESCRIPTION

The Feedback CHOP allows you to get the state of a CHOP as it was one frame or time slice ago. This allows you to connect CHOPs in circular loops without incurring the dreaded 'Infinite Recursion' error, because it simply copies its input without cooking it first.

POPs do this inherently, and some CHOPs like Lag and Filter look back in time (iterate) internally to the CHOP, but sometimes the output of a chain of CHOPs is needed as the input of the same chain one frame later.

For example, if you need the position or speed of an object from a frame ago in order to compute its position, displacement or speed at the current frame, you would select a CHOP containing those values and feed it into the Feedback CHOP. It will then output it a frame or time slice later.

Since its input is not recooked, it must be forcibly updated by following it with an exported CHOP to a displayed object.

## 22.2 PARAMETERS

**OUTPUT** */output*

Specifies how the input should be copied:

*Previous Channels at Previous Time*
Grabs the channels at prior time.

*Previous Channels at Current Time*
Grabs channels at current time.

*Last Sample at Current Time*
Uses the last sample received.

**DELTA TIME** */delta*

Time differential during feedback. If enabled, it adds a *dt* channel whose value is the elapsed time since the last cook. It is expressed in units determined by the Units parameter of the Common page.

## 22.3 STANDARD OPTIONS AND LOCAL VARIABLES

There are no local variables.

# 23 FETCH CHOP

## 23.1 DESCRIPTION

The Fetch CHOP imports channels from other OPs. It can grab either the regular control channels from any OP, or data channels that are output from CHOPs. Control channels are those channels associated with parameters of any OP. Data channels are only present in CHOPs, and are the channels output by CHOPs.

The Fetch CHOP can get the output channels from CHOPs in other CHOP networks.

Select the OP using menus, or by entering the path in the *Channels* field.

***Note: Fetching cannot be recursive. If a CHOP exports to an object's channel, and that object channel is fetched, the Fetch CHOP cannot feed to the CHOP that exports. This will produce an error message in the Fetch CHOP. If this occurs, you will have to lock the Fetch CHOP before exporting back to the object.***

## 23.2 PARAMETERS – SOURCE PAGE

### OP

The *OP* and *Channels* parameters determine where to obtain the channels.

The default gets all channels *( * )* of the object *geo1*, and the path is: */obj/geo1/**.
OP types are: *ch, comp, mat, obj, out, part* – which are: CHOPs, COPs, Materials, Objects, Output OPs and  s, respectively.

### CHANNELS

This field is appended to the specified OP (above) to complete the path. Wildcards (such as * and ?) and multiple entries are allowed.

An entry such as *t?* specifies all translate channels ( *tx, ty, tz* ).

Alternately, you can set all OP menus to *none* and specify the full path in *Channels*, such as: */obj/geo1/** .

#### specifying groups

You can specify groups in the Channels field using the @ symbol before the group name. For example:  */obj/@groupname*.

### FETCH

All OPs including CHOPs have OP Control Channels, which are the parameters in the dialog boxes of all Houdini OPs. CHOP Data Channels are output only by CHOPs. If you're not fetching from a CHOP, leave this menu set to *OP Control Channels*.

### SELECT

Fetched channels can be eliminated to include only control channels that have been created, or further, channels that have been created and are animating over the start/end time range. This menu applies to OP Control Channels only.

*All Channels* — All channels that match the *OP* and *Channels* path are included.

*Created Channels* — Only channels that have been created and appear in the channel list for the OP) are included.

*Non-Constant Channels* — Only created channels that are animating over the start/end range are included.

## 23.3 PARAMETERS – CHANNEL PAGE

### CHANNEL NAMES

Sets how the fetched channels are named.
A colon ( : ) is placed in the channel name where / is in the path.

*Channel Name* — Only use the name of the channel.

*OP and Channel Names* — Include the parent OP name with the channel name, separated by a colon ( : ).

*Full Path Name* — Use the full path as the name, with the parts separated by colons ( : ).

### CHANNEL RANGE

Indicates how much of the channel to fetch:

*Use Full Animation Range* — All of the animated range. If any of the specified channels are animating, there will be samples for each frame of the animation. Otherwise, if all the specified channels are constant, the CHOP length will be just one sample long.

*Use Current Frame* — Only the sample at the current frame.

*Use Start/End* — Specify the range below.

### START / END

Specify the beginning/end sub-section of the fetch. *Units* are set in the *Common* page.

### SAMPLING METHOD

Determines the sample rate to be used:

*New Rate (Same Index Range)*
Resample to new rate below.

*Resample From Maximum Rate*
Use the highest rate found.

*Resample From Minimum Rate*
Use the lowest rate found.

### SAMPLE RATE

The sample to use if *New Rate* is selected, or if no rates are found.

## 23.4 STANDARD OPTIONS AND LOCAL VARIABLES

- There are no local variables.

# 24 FILE CHOP

## 24.1 DESCRIPTION

This CHOP reads in channel and audio files for use by CHOPs.

### VALID FORMATS

The types of files that can be read into CHOPs include:

| | |
|---|---|
| `.chan` | Raw ASCII channel files; a row of numbers per frame. |
| `.bchan` | Raw PRISMS/Houdini binary channel files. |
| `.clip .bclip` | Houdini native CHOP clip files. |
| `.chn .bchn` | Houdini channel segment files. |
| `.aiff .aifc` | Audio files. |
| `.au .sf .snd .wav` | Audio files. |

The same files can be output from the menu on the CHOP tile by selecting "Save Data Channels". For a complete listing of all valid formats for CHOPs, see the *Formats* section.

### OTHER INPUT DEVICES

For Midi files (.mid or .midi), see the *MIDI In CHOP* p. 382. Some other input devices, (e.g. Puppetworks) can be read with the *Pipe In CHOP* p. 415.

Users can extend the file formats that are read/written by the File CHOP through the CHOP*io* and *CLIPio* files in *$HFS/houdini*. Use the *clchan* and *clchn* programs as examples of the input/output needed.

## 24.2 PARAMETERS

### CHANNEL FILE

The name of the file to load.

### RATE OPTIONS

| | |
|---|---|
| *No Change* | The sample rate is taken from the file. |
| *Override* | The sample rate is set to the number in the Sample Rate parameter, but the channels in the file are not resampled. |

*Resample* Resamples at the specified Sample Rate. Takes the channels and the sample rate found in the file, and resamples it to the desired sample rate. This prevents very large arrays from being used in memory unnecessarily.

### SAMPLE RATE

Samples per second, as utilized by the *Rate Options* parameter.

## 24.3 STANDARD OPTIONS AND LOCAL VARIABLES

- *Extend Options* p. 303

# 25 FILTER CHOP

## 25.1 DESCRIPTION

This CHOP smooths or sharpens the input channels. It filters by combining each sample and a range of its neighbour samples to set the new value of that sample. Each filter type uses its own weighting factors for the neighbouring samples. The *Filter Width* determines the number of neighbours to use.

This CHOP can filter both motion and sound, but other CHOPs are more appropriate for filtering sound (see the *Pass Filter CHOP* p. 403, *Band EQ CHOP* p. 315, and *Parametric EQ CHOP* p. 406).

## 25.2 PARAMETERS

### TYPE

There are seven types of filters:

*Gaussian* — This filter has a Gaussian (normal or "bell" curve) shape that smooths the channel. It acts as a low pass filter. The wider the filter, the lower the cutoff frequency, resulting in smoother data.

*Left Half Gaussian* — This produces a lag on the channel. If the input channels represent values over time, this filter is seen as only using samples back in time from the current sample. For time-data, this is more realistic as you can't look ahead in time. (Maybe some day.) It has a half-bell shape.

*Box* — This filter is box-shaped, meaning that each neighbor sample it uses has the same weighting factor. It can produce unwanted steps in the output channel because the effect of the samples at the extremes of the filter don't fade out as the window slides over the samples. It low-pass filters data, similar to the Gaussian filter.

*Left Half Box* — This filter produces a lag on the data, uses only samples back in time, and otherwise acts like a box filter.

*Edge Detect* — This filter detects "edges", sharp changes in the input channels. It acts as a high pass filter. As the filter width is increased, more low frequencies are added.

*Sharpen* — This filter sharpens all high frequencies. It is the sum of the edge detect result and the original data.

*De-spike* This filter removes "spikes" (samples more than 'Spike Tolerance' above or below the expected sample value). The filter width allows you to eliminate spikes that are several samples long. Wide filters will remove wide spikes (spikes of several samples) and small filters will only remove narrow spikes (one or two samples in length).

#### EFFECT

The extent to which the filter affects the channel (0 - not at all, 1 - maximum effect).

#### FILTER WIDTH

The amount of surrounding samples used in the calculation of the current sample. It is expressed in the Units.

#### SPIKE TOLERANCE

For the *De-spike* filter type, this is the amount that a sample can differ from its neighbours without being considered a spike.

#### NUMBER OF PASSES

The number of times the filter is applied to the channel.

## 25.3 STANDARD OPTIONS AND LOCAL VARIABLES

- *Scope and Channel Name Matching Options* p. 296
- *Units Options* p. 297
- *Local Variables* p. 306 – $C, $NC

## 25.4 SEE ALSO

- *Pass Filter CHOP* p. 403
- *Band EQ CHOP* p. 315
- *Parametric EQ CHOP* p. 406

# 26 FUNCTION CHOP

## 26.1 DESCRIPTION

This CHOP provides more complicated math functions than found in the Math CHOP: trigonometic functions, logarithmic functions and exponential functions. Since many of these functions can produce math errors, an error handling tab is provided for error handling and recovery.

Most of the functions require only one parameter, and they are applied as a unary operator to each input channel. Some functions take two parameters, and these require the use of the second input. The first parameter, X, is always a value from a channel in the first input. The second parameter, Y, is a value from a corresponding channel in the second input. Channels from each input are paired by name or index.

Errors can be handled by replacing the bad sample with a pre-defined value or by using the value of the previous sample. Alternatively, cooking can be aborted upon error for debugging networks.

## 26.2 PARAMETERS – FUNCTION PAGE

### FUNCTION

Which math function to apply to the channels. All of the functions are unary functions except for the binary functions 'Arctan (Input1/Input2)' and 'Input1 ^ Input2'.

### BASE VALUE *baseval*

The value of the base for 'Log base N' and 'Base ^ Input1'.

### EXPONENT VALUE *expval*

The value of the exponent for 'Input1 ^ Exponent'.

### ANGLE UNITS

For trigonometric functions, the angles can be measured in Degrees, Radians, or Cycles (0 to 1).

### MATCH BY

How to pair channels together from the two inputs for the binary functions, by name or by channel index.

## 26.3 PARAMETERS – ERROR PAGE

### ERROR HANDLING

How to correct samples with math errors:

*Abort With Error Message* Cooking aborts.

*Replace With Specified Values*
Values specified below.

*Use The Previous Value* Uses the last good result.

### + INFINITY VALUE *pinfval*

Value to use when an infinity error occurs. Caused by sinh(), cosh() and tan().

### - INFINITY VALUE *ninfval*

Value to use when a negative infinity error occurs. Caused by sinh() and tan().

### DOMAIN ERROR VALUE *domval*

Value to use when a domain error occurs. Caused by asin(), acos(), log10(), logN(), ln() and sqrt().

### DIVIDE ERROR VALUE *divval*

Value to use when a divide by zero error occurs. Caused by pow(x,y).

## 26.4 LOCAL VARIABLES

*none.*

# 27 GEOMETRY CHOP

## 27.1 DESCRIPTION

The Geometry CHOP uses a geometry object to choose a SOP from which the channels will be created. The channels are created from the point attributes of a SOP, such as the X, Y and Z of the point position.

The SOP can select a subset of the points using Point Groups. The set of attributes that are converted to channels are chosen using the names of the attributes seen on the Info popup of SOP tiles.

### SEE THE CHANNEL SOP

***Note:*** This CHOP works in tandem with the *Geometry > Channel OP* p. 491. Point data can be modified in CHOPs and then fed back to SOPs through the Channel SOP, therefore be sure to take a look at the Channel SOP description.

## 27.2 PARAMETERS – GEOMETRY PAGE

### OBJECT / SOP

Specifies which Object / SOP contains the geometry you want to fetch.

### GROUP

Only points within the specified group are Fetched. If blank, all points are fetched.

### METHOD

Choose whether to get the geometry for the current frame or for all frames:

*Static* — Gets the points for the current frame only, and builds one geometry point per sample of the CHOP. There is one channel per Attribute (The default is the P attribute which creates the tx, ty and tz channels).

*Animated* — Good for looking at point XYZ coordinates over the whole animation for SOPs whose shape animates. The CHOP cooks the input once per frame (of the CHOP's frame range), and create a set of channels for every point of the SOP. This can create a lot of channels! Start and End on the *Channels* page can restrict the frame range that gets created.

### ATTRIBUTE SCOPE

This selects the attributes of the SOP to acquire. By default, there are three channels for the XYZ position. The "P" attribute is the point position.

You can use any attribute. If you look at the SOP's info and there are other attributes, you can specify them. For example "uv" will get the three texture coordinates. The most common attributes are

| | |
|---|---|
| P | Point position (X, Y, Z) – 3 values |
| Pw | Point weight – 1 value |
| Cd | Point color (red, green, blue) – 3 values |
| Alpha | Point alpha – 1 value |
| N | Point normal (X, Y, Z) – 3 values |
| uv | Point texture coordinates (U, V, W) – 3 values |

See *Geometry Types > Attributes* p. 233 for a complete listing of Attributes.

### RENAME SCOPE

This parameter matches each channel acquired in the Attribute Scope. There must be one name per attribute value. By default, it translates the P attribute (position of the point) to tx, ty and tz channels. You can use any *Scope and Channel Name Matching Options* p. 296 as described in *Standard Options of CHOPs*.

### TRANSFORM OBJECT

If a transform object is specified, the point values will be represented relative to that object's origin and rotation.

### ORGANISE BY ATTRIBUTE

You can type 'pid' here – then all the channels get sorted properly.

## 27.3 SEE ALSO

- *SOPs > Channel OP* p. 491.

## 27.4 PARAMETERS – CHANNEL PAGE

See *Channel Page* p. 302 for a description of these parameters:

- *Start-End Options* p. 302
- *Extend Options* p. 303

## 27.5 STANDARD OPTIONS AND LOCAL VARIABLES

- *Extend Options* p. 303
- *Units Options* p. 297

# 28 GESTURE CHOP

## 28.1 PARAMETERS

Gesture records a short segment of the first input and loops this segment in time with beats from the third input. The second input defines the “listen” input.

When the first channel of the listen input goes above zero, the Gesture CHOP begins recording the first input's channels. While listen is on, the input channels are output exactly as is. When the listen is turned off, the recorded segment of the channels is processed (trimmed and blended). While listen is off, the recorded segment is looped continously.

The beat input should be a single periodic 0-1 ramp channel similar to the ones produced by the Beat and MIDI In CHOPs. The listen channel should be a single 0-1 on/off channel (ie, Keyboard or Logic CHOP).

The Gesture CHOP determines the number of beats that the listen was on for; this defines the period of the loop. If the beat frequency changes, the period will change with it.

The beat input is optional if the 'Fit Method' is set to 'None'. In this case, the recorded segment will be looped back with a period equal to the recorded length.

## 28.2 PARAMETERS

### FIT METHOD

Determines how to fit the recorded segment to the beats:

*None* — The beats are not used. The full segment is recorded and looped.

*Stretch To Beats* — The full segment is recorded and stretched to the beat period.

*Trim To Beats* — The closest beats to the start and end of the segment are selected, and the segment is trimmed, shifted or interpolated to fill the interval.

### FIT TO BEAT

The number of beats that the recorded segment spans can be automatically determined or fixed at a constant value:

*Auto Fit* — Fits to the closest number of beats.

*Auto Fit to Multiple of Beats* — Fits to the closest number of beats, as long as they are a multiple of the 'Number Of Beats' parameter. (i.e. for 2: 2,4,6,8,10,...)

*Auto Fit to Power of 2 Beats*

Fits to the closest number of beats that is a power of 2 (2,4,8,16,32...)

*Fit to Fixed Number of Beats*

Fit to the a fixed number of beats, specified below.

#### NUMBER OF BEATS */numbeats*

The fixed number of beats to fit the recorded segment to.

#### BLEND REGION */blend*

How much of the recorded segment to use as a blend region. The blend region is used to blend the beginning of the segment to the end so that a seemless loop is produced.

#### LISTEN BLEND */lblend*

When *Listen* is turned on or off, jumps can occur. This parameter provides a blend region that smooths them out.

#### INTERPOLATE SAMPLES

If on, recorded samples are interpolated when scaling occurs, otherwise the nearest sample is selected.

## 28.3 LOCAL VARIABLES

None.

# 29 HANDLE CHOP

## 29.1 DESCRIPTION

The Handle CHOP is the "engine" which drives Inverse Kinematic solutions using the Handle object. The role of the Handle CHOP is to generate rotation values for the bones which will bring their attached handles as close to their respective targets as possible.

The use of this methodology is best described with an example:

1. Increase the frame range to 10000. Create a Null object called 'target' and animate a few random positions for it to travel to.

2. Create a three bone IK chain with *No Kinematics*. Append a Handle object to the last bone.

3. In the Handle object set the Target menu to *Target*. This is the null object you just created.

4. The system is now specified. To actually start the IK enter a CHOP pane.

5. Place a Handle CHOP. In the *Source* field type in the names of the bones that were earlier created.

   ***Tip:*** In the Viewport, you can use the Select state to select the bones you created. Then in the Handle CHOP click the *Grab Source Form Selection* button. The names are entered automatically for you. You don't have to worry about being too picky when selecting objects this way, since non-bone objects are ignored.

6. Click the *Export* button on the Handle CHOP and click *Play*. You should now see the two-bone system chasing the animated null.

You can place any number of bones with any number of (possibly zero) Handles attached to the system. The following parameter description will now give a more detailed explanation of the functionality:

## 29.2 PARAMETERS

### SOURCE

Creates rx/ry/rz channels for each bone listed.

### GRAB SOURCE FROM SELECTION

Short-cut for entering bone names.

### FIXED

If you have entered bones which form a branch or should act as a unit, enter them here. An example may be two bones splitting at the shoulders. You want them to rotate, but only as a unit.

### GRAB FIXED FROM SELECTION

Short cut for entering bone names.

### ITERATIONS

Increasing this parameters gives a more accurate solution at the cost of cooking time. Preroll (only when precalculating a range of frames, SingleFrame turned off): This will cook the solution a given number of frames before the requested frame range. Max Angle Change: This will limit the delta angle degrees the bones can move in any given frame.  Use this to tame erratic behaviour.

## 29.3 SEE ALSO

- Handle Object

# 30 HOLD CHOP

## 30.1 DESCRIPTION

The Hold CHOP allows you to sample and hold the value of the first input. The second input controls the sampling. When the second input changes from 0 to 1, the first input is sampled. This value is held in the output until the second input changes from 0 to 1 again. Hold does not sample while the second input is 1, nor on the falling edge (1 to 0).

A common application for this CHOP is to grab the current value of a channel when an event occurs, so that value can be used until the event occurs again.

This CHOP can be time-sliced by enabling the *Time Slice* parameter. It can also take advantage of Minimal Time Slice Cooks in time slice mode (see the *Time Slice Preferences* dialog in the *Options* menu).

## 30.2 PARAMETERS

### SCOPE / SAMPLE RATE MATCH / UNITS

These parameters are covered in: *Common Page* p. 296

### TIME SLICE / UNLOAD / EXPORT PREFIX / GRAPH COLOR

These parameters are covered in: *Common Page* p. 296

## 30.3 LOCAL VARIABLES

None.

# 31 IK (INVERSE KINEMATICS) CHOP

## 31.1 DESCRIPTION

This CHOP calculates an inverse kinematics simulation for Bone objects.

It generates channels for bone objects based on a bone chain and an end affector. It can solve for the bone angles using a variety of different solver types.

## 31.2 PARAMETERS – KINEMATICS PAGE

### SOLVER TYPE

This parameter defines the method used to determine the motion of the Bone object when it, its ancestor, or its child are moved.

#### none

No solver. Use the original parameter values.

#### show rest position

No solver. Uses the rest angles.

This solver is used to show the rest position of the bones. It orients the bones at the rest position defined by the rest angles in the chain. These rest angles are used in the Inverse Kinematics solver. Therefore, this solver can be used as a helper utility to show you the rest positions of the chain.

#### show capture position

No solver. Uses capture angles.

This solver is used to display the rotation values in the capture angles parameter. These parameters are set in the Skeleton SOP when a "capture points" operation is performed, or by using the *Grab Capture Angles* menu. Like the *Show Rest Position* solver, this solver can be used to return a chain to its capture orientation.

#### inverse kinematics

Uses IK solver. The solution is uniquely defined by the end affector position. Constraint parameters are ignored.

The IK solver tries to position the bones such that the end of the last bone in the chain touches the origin of the object defined as the end affector. The shape of the chain is derived from the rest angles of each the bones in the chain.

In order to adjust the results of the IK solver, you can adjust the rest angles of the bones. Note that the IK solver's solutions are guaranteed to be continuous only for a given set of rest angles. If you animate the rest angles you may end up with chains that jump around.

When using kinematics, the rotations given by the kinematic solver are put into each bone's rotation value. In other words, the rotation values of each bone may seem to be animated even though the bone doesn't have any rotation channels. Adding rotation values to bone objects in a kinematic chain is not recommended, as those values will be over-written by the kinematic solver.

#### ik with constraints

Uses a modified IK solver. The solution depends on the previous solution. Constraint parameters are used to limit the range of motion.

#### follow curve

The bones are positioned such as to align themselves along the first curve in an objectas closely as possible. It is useful for animating things such as tails and spines.

### ROOT BONE

Where the chain starts. It is the first bone in the chain for which this CHOP should create solution channels.

### END BONE

If you specify a bone as the last bone in chain, then it is the last bone in the chain for which this CHOP should create solution channels.

### END AFFECTOR

For Inverse Kinematics and IK With Constraints solvers, select an object from this menu to use as the end affector when solving for the bone angles.

### TWIST AFFECTOR

For the Inverse Kinematics solver, this specifies an object that controls the twist orientation of the solution bone angles.

If you specify a twist affector, the entire bone chain will twist along the axis from the chain's root to its end-affector so that the first bone is pointing as much as it can at the twist affector. The twist value can be specified as well, and is applied on top of the twist generated for the twist affector.

### IK TWIST *ikwist*

For the Inverse Kinematics solver, this parameter specifies an additional twist angle to be applied to the solution bone angles.

### IK DAMPENING *ikdampen*

For the Inverse Kinematics solver, this parameter represents how easily the End Bone can be pulled off the End Affector as the bone chain is stretched out.

### CURVE OBJECT

For the Follow Curve solver, this parameter specifies the object that contains the curve geometry that the bone chain should follow.

## 31.3 PARAMETERS – CHANNEL PAGE

See *Channel Page* p. 302 for a description of these parameters:

- *Start-End Options* p. 302
- *Extend Options* p. 303

# 32 IMAGE CHOP

## 32.1 DESCRIPTION

This CHOP converts rows and/or columns of pixels in an image to CHOP channels. It generates one CHOP channel per span of pixels in the image. Pixels, rows, columns or rectangular regions can be extracted from the image.

There is an optional input which supplies UV coordinates to sample the image. The input CHOP must contain two channels, one for U and one for V. The first channel is always assumed to be U. The channels produced (red, green, blue and alpha) will be exactly as long as the input channel's length, with a value for each UV coordinate.

To import a CHOP back into a COP network, use the Channel COP.

## 32.2 PARAMETERS – IMAGE PAGE

### COP PATH

Specify the COP Network, and the COP here as the source from which the image should be obtained.

### FRAME

The frame number from which to obtain the image.
This defaults to $F – which always provides the current frame.

### RED / GREEN / BLUE / ALPHA

Creates a CHOP channel per scanline with the prefix specified here, starting at 0. For example, the default is “r” in the *Red* field. If the specified COP is a horizontal ramp that is five lines high, it creates five CHOP channels (r0 - r4) using the values of the red pixels in those scanlines.

### RGBA UNITS

Scales the output scale to lie in the range 0-1; 0-255; or 0-65535.

## 32.3 PARAMETERS – CROP PAGE

### CROP

Specifies what portion of the image to extract.

### UV UNITS

Specifies the units for the following four parameters.

### U START *ustart*

Starting point for sampling in U. Values outside the range of the image are determined by the image's extend conditions, in the *Extend* page.

### U END *uend*

Ending point for sampling in U.

### V START *vstart*

Starting point for sampling in V.

### V END *vend*

Ending point for sampling in V.

### INTERPOLATE

Determines the interpolation method when using *UV sampling with an input CHOP.*

## 32.4 PARAMETERS – EXTEND PAGE

These parameters determine what to do with parts of an image that exceed the bounds of the original image. You can select from the following extend conditions for each of these parameters using the menus:

| | |
|---|---|
| *Hold* | Use the first or last pixel value. |
| *Cycle* | Loop back to the other side of the image. |
| *Mirror* | Zig-zap back into the image. |
| *Default Color* | Use a default color specified below. |

### IMAGE LEFT

The image extend conditions when sampling the image with U less than 0.

### IMAGE RIGHT

The image extend conditions for U greater than 1.

### IMAGE BOTTOM

The image extend conditions for V less than 0.

### IMAGE TOP

The image extend conditions for V greater than 1.

### DEFAULT COLOR

The colour to use when outside the bounds of the image, and the Default Color extend condition is set.

## 32.5 STANDARD OPTIONS AND LOCAL VARIABLES

• *Units Options* p. 297

## 32.6 SEE ALSO


- *Geometry CHOP* p. 352
- *Particle CHOP* p. 410
- *Shuffle CHOP* p. 442

# 33 INTERPOLATE CHOP

## 33.1 DESCRIPTION

This CHOP treats its multiple-inputs as keyframes and interpolates between them. The inputs are usually single-frame CHOPs like those produced by a Constant CHOP. The Interpolate CHOP first sorts the input CHOPs in time (without shifting them) and interpolates between them to fill the gaps.

The number of channels in the output is the same as the number of channels in the first input.

If a channel is missing in an input, and *Match By* is set to *Channel Name*, it is treated as if there is no keyframe at that frame for that channel, and the interpolation occurs between CHOPs before and after that frame.

When the graph is in Bar Display, and you click on the black lines of the Interpolate CHOP bar which represent the inputs of the CHOP, you can drag the line to change the time of that input. It goes to that input and changes the value if its "start" parameter, so you can re-time the keyframes.

When you Shift-click on a black line, the current CHOP changes to the corresponding input, so you can edit its values. See *Keyboard Short-cuts for the CHOP Graph* p. 163 for more options.

## 33.2 PARAMETERS

### SHAPE

The shape of the interpolation curve:

| | |
|---|---|
| *Linear* | A straight line. |
| *Ease In* | Exponential rise. |
| *Ease Out* | Exponential fall. |
| *Ease In Ease Out* | Half cosine blend. |
| *Cubic* | A cubic spline. |
| *Add* | Overlapping parts, if any, are added. |

### OVERLAP PRIORITY

If an input is not a single frame, and if there are overlaps in the input CHOPs, an option is used to resolve the conflict.

| | |
|---|---|
| *Average Overlap* | Combine the conflicting channels by averaging them. |
| *First Segment has Priority* | Use the segment with the earliest start time. |
| *Last Segment has Priority* | Use the segment with the latest start time. |

## 33.3 STANDARD OPTIONS AND LOCAL VARIABLES

- *Match By Options* p. 304
- There are no local variables.

# 34 KEYBOARD CHOP

## 34.1 DESCRIPTION

This CHOP receives ASCII input from the keyboard, and outputs channels for up to nine keys. It creates a single-frame channel representing the current state of each key.

The channel for a key is created by putting a channel name string into one of the *Name* parameters, and selecting that key from the corresponding *Key* menus. Each key event gets routed to one destination in order of priority:

1) Input field (only if selected)
2) CHOPs (only if used)
3) Anywhere else.

A textport command states where each key is routed (wrt Keyboard CHOPs).

### INTERCEPT MODE

The Keyboard CHOP can only be used while in "Intercept" mode.
To enable Intercept mode, you need to have the *Scroll Lock* on. The Scroll Lock key is located at the top right of keyboard after the row of Function keys.

When in Intercept mode, the Houdini Playbar will change to an orange color and all keyboard input will be redirected to any Keyboard CHOPs. This means you cannot access any regular keyboard short-cuts (such as View state keyboard short-cuts, edit OP parameters, or use the Textport) while Intercept is on.

To use the numbers on the Keypad, you need to have NmLk turned on.

The Mouse and Keyboard CHOPs are often connected to the Position and Active inputs of the Record CHOP to enable the recording of channels.

## 34.2 PARAMETERS

### TYPE

The *Type* menus determine which of the several key modes is chosen:

| | |
|---|---|
| *Momentary* | This is simply the up/down state of a key.<br>0 is up and 1 is down. |
| *Toggle* | This toggles the key state. Push once and it goes on (1), push again and it goes off (0). |
| *Count* | This increments the channel value by one every time the key is pressed. |
| *Pulse* | This produces a one-frame pulse when a key is pressed. |
| *Time* | This returns the length of time that the key has been |

held down. It only works while the Playbar is playing.

## 34.3 PARAMETERS – KEYS 1 TO KEYS 3 PAGES

### NAME 1

The name of the channel that monitors the selected key. If blank, a channel will not be created.

### TYPE 1

The Type of key: *Momentary*, *Toggle*, *Count*, *Pulse* or *Time*. (see descriptions of parameters, above).

### KEY 1

The Key selector, where valid keys are:

- Numbers 0 - 9
- Letters A - Z
- Keypad 0 - 9 (must have NmLk on)

## 34.4 PARAMETERS – CHANNEL PAGE

This parameters in this page set the Sample Rate and the Extend Conditions.

See the descriptions in *Standard CHOP Options > Channel Page* p. 302.

## 34.5 STANDARD OPTIONS AND LOCAL VARIABLES

- *Extend Options* p. 303
- *Units Options* p. 297
- There are no local variables.

# 35 LAG CHOP

## 35.1 DESCRIPTION

This CHOP adds lag and overshoot to channels. It can also limit the velocity and acceleration of channels. Lag is an effect that slows down rapid changes in the input channels. Overshoot amplifies the changes in the input channels.

Two values exist for each parameter. For example, in the Lag effect, when the input channel value is rising, the first lag parameter is used, and when the channel value is decreasing, the second lag parameter is used. This can give a quick rise, and a slow fall. But lag up and down are often kept at the same value.

The *Spring CHOP* p. 452 provides some similar effects.

## 35.2 PARAMETERS

### METHOD

The method by which lag is applied to the channels.

*Lag Value* — Up is defined as an increase in values of the input channels, and down is a decrease in values of the inputs.

*Lag Amplitude* — Up is defined as an increase in amplitude (moving away from zero) and down is defined as a decrease in amplitude (moving towards zero) of each input channel.

*Lag Magnitude* — All the input channels are treated as components of one vector, and each operation is applied to the vector as a whole. Only the first parameter in Lag, Overshoot, Clamp Slope and Clamp Acceleration applies in this mode.

### LAG

Applies a lag to a channel. The first value is for lagging up, and the second is for lagging down. It is approximately the time that the output follows 90% of a change to the input.

### OVERSHOOT

Applies overshoot to a channel. The first value is for overshoot while moving up, and the second is for overshoot while moving down.

### CLAMP SLOPE

Clamps the slope (or velocity) to lie between the values listed in *Max Slope* below. Slope is expressed as value/Units.

### MAX SLOPE

The values to limit the slope of the channels between.

### CLAMP ACCELERATION

Clamps the acceleration to lie between the values listed in *Max Acceleration* below. Acceleration is expressed as *value/(Units*2)* .

### MAX ACCELERATION

The values to limit the acceleration of the channels between.

### COOK TO CURRENT FRAME

If enabled, only the part of the CHOP between the time of the last cook and the current frame is updated.

## 35.3 STANDARD OPTIONS AND LOCAL VARIABLES

- *Scope and Channel Name Matching Options* p. 296
- *Units Options* p. 297
- *Local Variables* p. 306 – $C, $NC, $I

# 36 LIMIT CHOP

## 36.1 DESCRIPTION

This CHOP provides a variety of functions to limit and quantize the input channels.

Limiting a channel causes all its values to lie within a range. Several different methods are available for limiting:

- *Off* – Do not limit the values.
- *Clamp* – Simply cut the channel value off if it is out of the Maximum/Minimum range, and replace it with the Maximum or Minimum limit value.
- *Loop* – Continue the channel at the other end of the interval.
- *Zigzag* – Mirror the values back inside the interval.

Quantizing a channel value snaps its values to the closest allowable value (the "quantized values"). Quantizing methods are: Floor, Ceiling, and Round.

Quantizing a channel index is like quantizing in time, and acts as a sample and hold mechanism. The channel is sampled at a quantized index, and held at that value until the next quantized index at which time the value takes on the input value at that point.

## 36.2 PARAMETERS – LIMIT PAGE

### TYPE

The type of limit function to use:

| | |
|---|---|
| *Off* | Disabled. |
| *Clamp* | Clamps values. |
| *Loop* | Loops values. |
| *Zigzag* | Values which go back and forth. |

### MINIMUM

The minimum value the output channel can have.

### MAXIMUM

The maximum value the output channel can have.

### POSITIVE ONLY

Takes the absolute value of the channel, making all negative values positive.

### NORMALIZE

Scale and offset the channel so that it lies between -1 and +1.

## 36.3 PARAMETERS – QUANTIZE PAGE

### QUANTIZE VALUE

Selects the quantization method to use:

| | |
|---|---|
| *Off* | Disabled. |
| *Floor* | The Floor quantization function rounds a value to the closest quantized value below it. |
| *Ceiling* | The Ceiling function rounds a value to the closest quantized value above it. |
| *Round* | The Round function rounds a value to the nearest quantized value. |

### VALUE STEP

The increment between quantized values.

### VALUE OFFSET

The offset for quantized values, to allow steps to not lie at zero, the default.

### QUANTIZE INDEX

Selects whether to quantize the index relative to the sample 0, or the start index of the CHOP.

### INDEX STEP

The increment between quantized indices, in seconds, frames or samples.

### INDEX OFFSET

The offset for quantized indices.

## 36.4 STANDARD OPTIONS AND LOCAL VARIABLES

- *Scope and Channel Name Matching Options* p. 296
- *Local Variables* p. 306 – $C, $NC

# 37 LOGIC CHOP

## 37.1 DESCRIPTION

This CHOP converts channels of all its input CHOPs into binary channels and combines them using a variety of logic operations. 0 is considered off, and 1 is on.

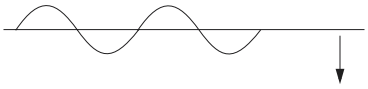
## 37.2 PARAMETERS – LOGIC PAGE

### CONVERT INPUT

This menu determines the method to convert inputs to binary:

*Off When Zero* — Returns a logical 0 when channel amplitude is zero; Non zero values return 1.

*Off When Zero or Less* — Returns a logical 0 when a channel amplitude is zero or less; and 1 when values are positive.




### CHANNEL PRE OP

Once converted by the Convert Input stage, *Channel Pre OP* defines a unary operation on each input sample:

*Invert* — Changes 0 to 1; and 1 to 0.

*Toggle* — Causes each 0 to 1 transition of the input channel to switch the current state between 0 and 1.

*Radio Button* — Only one channel per input can be on at once. If another channel turns on, the previously "on" channel is turned off.

*Last Two On* — This is like Radio Button, but it keeps up to two channels on. If followed by a Lag CHOP, it is useful for blending between pairs of poses.

*Rising Edge* — On for one sample only, at each place where a channel goes from off to on.

*Falling Edge* — On for one sample only, at each place where a a channel goes from on to off.

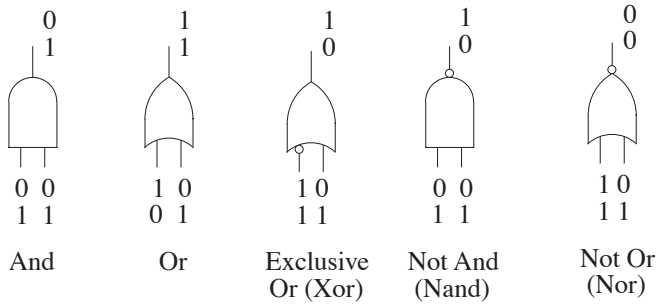
### COMBINE CHANNELS

Takes the first input and combines its channels, then the second input and combines its channels, and so on.

### COMBINE CHOPS

Combine CHOPs combines And the first channel of each CHOP, the second channel of each CHOP, etc.. Channels between inputs can be combined by number or name.

**combining (logic) operations are:**




| | |
|---|---|
| *And* | On if all inputs are on, otherwise off. |
| *Or* | On if at least one input is on, otherwise off. |
| *Exclusive Or* | On if one input on, but not more than one. |
| *Not And* | Off if all inputs are on, otherwise on. |
| *Not Or* | Off if at least one input is on, otherwise on. |
| *Equivalence* | On if inputs are equal (all on or off), otherwise off. |
| *First Channel On* | Gives the index of the channel that was the first on, or -1 if no channels are on. |
| *Last Channel On* | Gives the index of the channel that was the last on, or -1 if no channels are on. |

### MATCH BY

Channels are matched between inputs by Channel Name or Channel Number.

### ALIGN

Inputs that don't start at the same frame can be aligned. Se the section, *Align Options*.

### COOK TO CURRENT FRAME

Logic has two ways of cooking, set with *Cook To Current Frame*. If on, only the part of the CHOP between the time of the last cook and the current frame is updated. When off, each cook of the Logic CHOP cooks the whole interval.

## 37.3 PARAMETERS – SCRIPT PAGE

### OFF TO ON

The script executed when an output channel switches from off to on, called at the first "on" frame. You can use variables like $C and $I in the script commands to identify which channel and frame went off to on. These scripts are often used during realtime execution of CHOPs by running scripts with *Cook to Current Frame* on.

### WHILE ON

The script executed when an output channel is on. Called once per channel that is on, each frame.

### ON TO OFF

The script executed when an output channel switches from on to off, called at the first "off" frame.

### WHILE OFF

The script executed when an output channel is off.
Called once per channel that is off, each frame.

## 37.4 RESULT OF CHANNEL AND CHOP INPUT OPERATIONS

| Combine Channels | Combine CHOPs | Result (Number of Channels in Output) |
|---|---|---|
| Off | Off | Total number of channels in all inputs |
| Off | On | Number of channels in the first input |
| On | Off | Same as the number of inputs |
| On | On | One channel |

## 37.5 STANDARD OPTIONS AND LOCAL VARIABLES


- *Scope and Channel Name Matching Options* p. 296
- *Sample Rate Match Options* p. 297
- *Align Options* p. 303
- *Local Variables* p. 306 – $C, $NC, $I, $V

# 38 LOOKUP CHOP

## 38.1 DESCRIPTION

This CHOP uses a channel in the first input to index into a lookup table in the second input, and output values from the lookup table. The first input is the Index input, and should contain a single channel to use as an index reference. The second input is the Lookup Table, and can contain any number of channels.

A 0 to 1 in the Index range means that a 0 in the Index (first) input maps to the beginning (left) of the lookup table, and 1 in the Index (first) input maps to the end (right) of the lookup table (second input).

The output CHOP is the same length as the Index input. It is the same number of channels as the Lookup Table input, and has the same channel names as the Lookup Table input. The output is filled with the data extracted from the lookup tables. Indexes that fall between two lookup table samples are interpolated. The lookup table can be sampled outside its range. The *Extend Conditions* are used in this case.

The sample rate of the output is the sample rate of the Index input.

The Lookup CHOP can be used to fetch values from a color lookup table, where a single index produces red, green and blue channels. It can also be used for rolloff or decay tables, where it specifies how much a parameter drops with distance.

## 38.2 PARAMETERS

### INDEX RANGE

The *Index Range* maps the index channel's values to the lookup table's start and end. The first parameter represents the start of the lookup table. When the index channel has this value, it will index the start of the lookup table. The second parameter represents the end of the lookup table and behaves in the same way.

## 38.3 STANDARD OPTIONS AND LOCAL VARIABLES

- There are no local variables.

# 39 MATH CHOP

## 39.1 DESCRIPTION

This CHOP allows you to perform a variety of arithmetic operations on and between channels. Channels of a CHOP can be combined, and several CHOPs can be combined.

### UNARY OPERATIONS

Unary operations are performed on individual channels. Unary operations are:

| | |
|---|---|
| *Off* | Don't do anything to the channel. |
| *Negate* | Take the negative value of each sample of the channel. |
| *Positive* | Make negative values of the channel positive (absolute). |
| *Root* | Take the square root of all values in the channel. |
| *Square* | Square all the values in the channel. |
| *Inverse* | Take the inverse (1/x) of all values in the channel. |

### COMBINE CHANNELS

Operations between channels can be done within an input or between inputs. The operations are done on a sample by sample basis:

| | |
|---|---|
| *Off* | Don't combine the channels. |
| *Add* | Sum all the channels. |
| *Subtract* | Subtract all the channels from the first. |
| *Multiply* | Take the product of all the channels. |
| *Divide* | Divide the first channel by all the rest. |
| *Average* | Take the average of all the channels. |
| *Maximum* | Take the maximum value of all the channels. |
| *Minimum* | Take the minimum value of all the channels. |
| *Length* | Assume the channels are a vector and compute its length. |

## 39.2 PARAMETERS – OP PAGE

The four steps are performed in the following order:

### CHANNEL PRE OP

A menu of unary operations (as described above) that are performed on single channels as they come in to the Math CHOP.

### COMBINE CHANNELS

A menu of operations that is performed between the channels of a single input CHOP, for each input.

### COMBINE CHOPS

A menu of operations that is performed between the input CHOPs, combining several CHOPs into one.

### CHANNEL POST OP

A menu of more unary operations that is performed on the channels resulting from the above operations.

### MATCH BY

Match channels between inputs by name or number.

### ALIGN

How to align channels from different inputs. See *Standard Options of CHOPs > Align Options* p. 303.

## 39.3 PARAMETERS – MULT-ADD PAGE

The three steps are next performed in the following order:

### PRE-ADD

First, add value to each new channel.

### MULTIPLY

Then multiply by this value.

### POST-ADD

Then add this value.

## 39.4 PARAMETERS – RANGE PAGE

This is the final step prior to output.

### FROM RANGE / TO RANGE

Another way to multiply/add. Converts from one low-high range to another range.

## 39.5 STANDARD OPTIONS AND LOCAL VARIABLES

- *Align Options* p. 303
- *Match By Options* p. 304
- *Scope and Channel Name Matching Options* p. 296
- *Sample Rate Match Options* p. 297
- *Local Variables* p. 306 – $C, $NC, $I, $V

# 40 MERGE CHOP

## 40.1 DESCRIPTION

This CHOP takes multiple inputs and merges them into the output. All the channels of the input appear in the output. The channel order is the channels of the first input followed by the channels of the second input, and so on.

Channel names may conflict, but a channel is renamed according to the *Duplicate Names* menu.

## 40.2 PARAMETERS

### ALIGN

The *Align Options* handle cases where multiple input CHOPs have different start or end times. All channels output from a CHOP share the same start/end interval, so the inputs must be treated with the Align Options as described in: *Standard Options of CHOPs > Align Options* p. 303.

### DUPLICATE NAMES

When channels of the input CHOPs have the same name, this menu determines what to do.

*Unique Names* — Include all the input channels, and when there is a conflict of name, make the new channel's name similar but unique by adding digits to the name.

*Keep First* — Include the channel of the first input CHOP which the name appears in, and discard any further channels with the same name.

*Keep Last* — Include the channel of the last input CHOP which the name appears in, and discard any other channels with the same name.

## 40.3 STANDARD OPTIONS AND LOCAL VARIABLES

- *Align Options* p. 303.
- *Sample Rate Match Options* p. 297
- There are no local variables.

# 41 MIDI IN CHOP

## 41.1 DESCRIPTION

The MIDI In CHOP reads Note events, Controller events, Program Change events, and Timing events from both MIDI devices and files.

For a quick start, see: *User Guide > CHOPs > Using MIDI in Houdini*.

The MIDI In CHOP receives MIDI events from MIDI devices connected to the serial port, reads MIDI events internal to the workstation (i.e. the built-in software synth), and interprets musical scores in MIDI files. MIDI data can be recorded in real time or recorded to the Houdini timeline play bar's time. Supported MIDI events are:

- Note On, Note Off
- Polyphonic Aftertouch
- Channel Pressure
- Program Change
- Control Change (MIDI controller devices)
- Pitch Wheel
- Timer Events including beat pulses
- Bar Messages
- Start, Stop, Continue
- Song Position Pointer
- System Exclusive Messages

MIDI events arriving on separate MIDI channels may be recorded on separate CHOP channels. Also, any number of MIDI CHOPs can read from the same or different sources. When Houdini receives a MIDI *Start*, *Stop* or *Continue* event. it affects the Playbar.

***Note:*** The values of the MIDI inputs are saved into the Houdini .hip file, and are restored when the file is reloaded. The physical controllers may be in a different position when the .hip is restarted, causing the values to jump when the controllers are moved. This is unavoidable.

## 41.2 PARAMETERS – SOURCE PAGE

### MIDI SOURCE

The CHOP will automatically detect all running interfaces upon cooking, and list them here.

### MIDI FILE

If reading a file, the name of the MIDI file to read.

### READ ENTIRE MIDI FILE

If enabled, the entire MIDI file is read. Otherwise, the *Start* and *End* parameters on the *Channel* page determine the segment of the file to read.

### MIDI CHANNEL

The CHOP may read from any number of MIDI channels, numbered 1-16. Ranges and multiple entries are supported (i.e. "`1 4 6`", "`1-7 12`", "`1-5:2`").

If *Channel Prefix* is left blank, then the input streams from multiple MIDI channels will be combined into one set of CHOP channels (i.e., a "note 64 on" event on channel 12 followed by a "note 64 off" event on channel 8 appears in the note 64 CHOP channel as a single note).

### CHANNEL PREFIX

When recording from multiple MIDI channels, putting a string like "`ch`" in this parameter causes the MIDI channel to be split into separate CHOP channels per MIDI channel. Otherwise the MIDI channels are all merged into one set of CHOP channels.

### ECHO MESSAGES TO TEXTPORT

This displays a raw log of all incoming MIDI messages to the shell. This is used to diagnose what MIDI events are coming into the workstation. Often you don't know what channel, note numbers or controller devices are generating events, if any. Turn on the Graph flag and play forward. Look at the shell that started Houdini and watch the messages as you modify MIDI devices.

## 41.3 PARAMETERS – RECORD PAGE

### RECORD METHOD

*Tie to Time Line* — The current time is taken as the record time. If the play rate of the doesn't match real time, the data will be stretched or compressed. This mode is preferable when using the MIDI device to control animation.

*Single Frame* — This only outputs the current value of notes, controllers, etc.. It does not retain prior values. It creates a CHOP at frame 1 always, so the CHOP causes cooking only when MIDI events come in.

*Current Frame* — Same as "Single Frame", but the CHOP frame is always at the current frame, so the CHOP changes every frame, causing other CHOPs to recook every frame.

*Time Line Independent* — Recording occurs in real time, independent of the current frame of the Houdini Playbar. Playback can be stopped. The graph will automatically update every half second while recording. This mode has the most accurate timing.

### RECORD CONTROL

*Manual* — Pressing the Record button (next parameter) will begin recording, and pressing it again will stop recording.

*MIDI Start/Stop Signals* — A start signal from the MIDI device is used to start recording, and a stop signal stops recording. Useful for synchronized recording. Once you have recorded data that you wish to keep, switch Record Control back to Manual to avoid accidental recording by future signals.

### RECORD

This parameter is used as a button to start and stop recording into the CHOP channels..

### RECORD POSITION

When the Record Method is set to Time Line Independent, you may start recording at any time after the starting time. This parameter has an associated handle. In combination with the Scope on the *Common* page, this can be used to layer more recordings on the CHOP channels.

### RESET

Sets the CHOP lengths to one sample, erasing the data in the channels.

## 41.4 PARAMETERS – NOTE PAGE

### NOTE NAME

Put an "n" in here to generate channels for note events. It is the base name of the CHOP channel used to record notes. If blank, notes are ignored. If the *Note Output* parameter is set to *Separate Channels*, then CHOP channels will be created for each note in the Note Scope, of the form: *notename* | *notenumber* (i.e. "n64").

### NOTE SCOPE

The scope of notes to record. Multiple ranges and notes can be recorded (i.e., "`50-60`", "`64 65 66 70-80`").

### NOTE OUTPUT

| | |
|---|---|
| *Separate Channel* | Each note number gets its own CHOP channel. |
| *One Multiplexed Channel* | One CHOP channel per MIDI channel is used for all notes. The value in the note channel is the number of the note currently playing (in the case of multiple notes playing, the most recent note is selected). |

### VELOCITY

| | |
|---|---|
| *Off* | Velocity is not recorded. |
| *Note Amplitude* | The velocity is recorded as the amplitude of the note. Only valid when Note Output is set to *Separate Channels*. |
| *Separate Channels* | Velocity is recorded in separate CHOP channels. Each note channel will have a corresponding velocity channel. |

### VELOCITY NAME

When Velocity is set to *Separate Channels*, this parameter is the base name of the Velocity CHOP channel (try "v"). If blank, no velocity channels will be recorded.

### AFTERTOUCH NAME

The base name of the polyphonic Aftertouch CHOP channels. One aftertouch channel is created for each note in the Note Scope. If blank, no aftertouch channels will be created.

### PRESSURE NAME

The name of the Channel Pressure channel. If multiple channels are being recorded, all channel pressure changes of interest will be recording on this CHOP channel. If blank, this channel will not be created.

### PROGRAM CHANGE

The name of the Program Change CHOP channel. All program change messages will be recorded onto this channel. If blank, this channel will not be created.

## 41.5 PARAMETERS – CONTROL PAGE

### PITCH WHEEL NAME

The name of the Pitch Wheel CHOP channel. Pitch wheel values range from -1 to +1. If blank, this channel will not be created. Put "p" in here to generate a channel.

### CONTROLLER NAME

The base name of the Control Change CHOP channels. The channel names are appended with the controller index (0-127). If blank, control changes will not be recorded. It typically contains "c".

### CONTROLLER TYPE

There are 128 different controllers available. By choosing *By Index Only*, you can specify any number of controllers in the *Control Index* parameter. Otherwise, you can select a controller from the list from this menu. Some controllers have multiple instances, such as *Sound Controllers 1-10*. Selecting a controller with multiple instances allows you to use the *Control Index* parameter to select which instances you want. Any invalid control indices will be ignored.

### CONTROLLER INDEX

Used to select controllers by number, or multiple controllers by ranges. When in *By Index Only* Controller Type mode, you can select up to the full 128 controllers or sub-ranges thereof (i.e. "`1-10`", "`2 34 70`", "`1-32 70-80:2`").

### CONTROLLER FORMAT

Some controllers can be paired together to form 14 bit controllers, rather than the normal 7 bit controllers (controller indices 0-31, 98 and 100). Selecting 14 bit Controllers interprets a pair as one 14 bit controller. Otherwise, they are interpreted as separate 7 bit Controllers.

### NORMALIZE

Controller values can be normalized for convenience:

| | |
|---|---|
| *None* | When Normalize is Off, 7 bit controllers will return values in the range 0-127 and 14 bit controllers will return values in the range 0-16383. |
| *0 to 1* | These values are normalized to lie in the range 0 to 1. |
| *-1 to 1* | These values are normalized to lie in the range -1 to +1. |
| *On/Off* | Several of the controllers are simply on/off states. When *Convert to On/Off* is enabled, values greater than or equal to 64 will be interpreted as "1" and all values less than 64 will be interpreted as "0". |

## 41.6 PARAMETERS – SYS PAGE - COMMON SUB-PAGE

### TIMER PULSE NAME

The name of a channel that pulses on MIDI beats. Most MIDI devices can send a MIDI timer signal. This is the name of a CHOP channel that will record this timing signal. If blank, the channel will not be created.

### TIMER RAMP NAME

The name of a channel that ramps from 0 to 1 between MIDI beats. Uses same MIDI timer signal as Timer Pulse Name.

### TIMER PERIOD / START

The names of two channels that are constant. The first channel is the MIDI pulse period, the time between pulses. The second channel is the time of the start of a pulse. These can be used to feed another CHOP that creates a reference start/end, such as what the Beat CHOP outputs.

### TICKS PER BEAT

Defines the number of timer pulses that make up one MIDI beat. For example, if it is set to 24, every 24th timer pulse is recorded as one pulse and ramp of the Timer Pulse and Ramp channels.

### BAR MARKER NAME

The name of a channel that ramps from 0 to 1 between MIDI bars. Uses same MIDI timer signal as Timer Pulse Name. After getting a Bar Marker event, the ramp does not start until the next tick event.

### BAR PERIOD / START

The names of two channels that are constant. The first channel is the MIDI bar's period, the time of one bar of music. The second channel is the time of the start of a bar. These can be used to feed another CHOP that creates a reference start/end, such as what the Beat CHOP outputs.

### BAR MESSAGE

To mark the start of a bar, a message can be sent in the MIDI stream. When it is received, the CHOP waits until it sees a MIDI Timer Pulse message (F8) before it starts the bar. The bar continues until the next Bar Message / Timer Pulse message.

The Bar Message has the same pattern-matching string as a System Exclusive message, as seen under the Exclusive1 to 3 pages.

### SONG POS NAME

The name of the Song Position pointer channel. This channel will contain the value of the last received song position.

## 41.7 PARAMETERS - SYS PAGE, EXCLUSIVE1 TO 3 SUB-PAGES

### CHANNEL N / MESSAGE N

Houdini can turn System Exclusive messages into CHOP channels. Each sub-page has enough room for four sysex messages and corresponding channel names.

The Exclusive sub-page's Message parameters are compared against the MIDI messages coming into the MIDI interface, and the Midi In CHOP tries to match them.

Example. Assume the following appears in the Message 1 parameter of the Exclusive 1 sub-page of the Sys page.:

```
F0 7F 01 * v F7
```

These numbers are in hexadecimal. Sysex starts with F0 and ends with F7. Each MIDI byte is separated by a space in the message string. A * matches any byte. A v means that the CHOP interprets the bytes as a value that should be put in the channel named by the Channel Name 1 parameter.

## 41.8 PARAMETERS – CHAN PAGE

This is similar to other generators' *Channel* page. However, there are a few points to consider on how they affect the Midi In CHOP.

### START

Defines where recording begins. In "Tie to Time Line" mode, any events received before the start time will be ignored. In "Time Line Independent" mode, recording will start at this point and continue on (not looping back). If reading from a MIDI File, Start/End will determine the start of the segment to read.

### END

Defines the end of the segment to read for MIDI Files.

### SAMPLE RATE

Defines the sample rate of this CHOP. If the sample rate is too low, a rapidly changing input may be misrepresented.

***Note:*** If the sample rate is too low, you may miss MIDI events. A note event may set a sample value to 1, and then the next note event less than 1/30 second later can set it to 0 on the same sample in the CHOP channel. So the event will be missed. Make the sample rate higher, like 600, to catch these events, or make sure that your on-off events coming in are a minimum time separation, as can be achieved through Opcode's MAX.

## 41.9 PARAMETERS – COMMON PAGE

### SCOPE

Scoped channels will be overwritten upon recording; tracks that are not scoped will not be overwritten (they will also not be removed). Altering the scope and re-recording is a good way to layer effects.

See *Standard Options of CHOPs > Scope and Channel Name Matching Options* p. 296.

## 41.10 STANDARD OPTIONS AND LOCAL VARIABLES


- *Scope and Channel Name Matching Options* p. 296
- *Units Options* p. 297
- *Extend Options* p. 303

- There are no local variables.

# 42 MIDI OUT CHOP

## 42.1 DESCRIPTION


The MIDI Out CHOP sends MIDI events to any available MIDI devices. These devices can be other software programs (midisynth) or devices attached to the serial ports. Channels are used to control the sending of the MIDI events. The channels are evaluated over the last time slice (from the last framebar position to the current).

An event is sent every time a channel changes its value during this slice. All timing is preserved, as long as the framebar is running in realtime. Channels are mapped to events by their name. Events like notes, controllers and velocities must be followed by the note/controller number (n65, c7). If the number is left off a note event, the note number is the value of the channel. Other events, which are sent to the entire channel, do not need a trailing number (pc, pw). The channel prefix can be used to identify the MIDI channel the event should be sent on (i.e. “ch1n45” assigns that Houdini channel to note 45 messages on MIDI channel 1).

## 42.2 PARAMETERS – DEST PAGE

| | |
|---|---|
| *MIDI Destination* | Where the MIDI events are sent to. |
| *MIDI File* | The filename of the output MIDI file. |
| *Write MIDI File* | Writes all the data to a MIDI file. |
| Send Current MIDI Events | Sends the current state of all incoming channels out as MIDI events. |
| *MIDI Channels* | The MIDI channel(s) to output to. If a channel prefix is not given, all channels will receive the same events. |
| *Channel Prefix* | The prefix string that all input channels must have in order to extract the channel number from their name (i.e. "ch1note44", with a channel prefix of "ch"). |
| *Echo Messages To Textport* | If enabled, all MIDI events are logged in the textport as they occur. |

## 42.3 PARAMETERS – OUTPUT PAGE

| | |
|---|---|
| *Pre-Queue Time* | MIDI events are generated and queued this amount of time ahead of the current frame. |
| *Delay Time* | All events will be played later by this amount of time. |
| *Max Jump Time* | If the Playbar increases by this amount or more since the last cook, MIDI events in that interval will be skipped. |

*Synchonize* — If Houdini could not cook in time to generate the next set of MIDI events, it recovers by one of two methods. Missed events are either skipped or output immediately.

*Automatic Note Off* — 'All Note Off' events can be sent upon the start and/or end of the output.

*All Notes Off* — Sends an All Notes Off message to all MIDI channels.

## 42.4 PARAMETERS – NOTE PAGE

*Note Name* — The base name of the note channels. If input channels have a number after the name, it is assumed to be the note number. If not, the channel value is assumed to contain the note number.

*Velocity Name* — The base name of the velocity channels.

*Aftertouch Name* — The name of the aftertouch channel.

*Pressure Name* — The name of the channel pressure channel.

*Normalize* — Channel values in the range 0-1 are mapped to 0-127.

*Pitch Wheel Name* — The name of the pitch wheel channel.

## 42.5 PARAMETERS – CONTROL PAGE

*Controller Name* — The base name of the controller channels.

*Controller Format* — Sends 7 or 14 bit controller events.

*Normalize* — Maps channel values from different ranges to 0-127.

*Program Change* — The name of the program change channel.

## 42.6 PARAMETERS – SYS PAGE

*Bar Ramp Name* — Clock ticks frequency is determined by the period of the ramp. The ramp must be 0 to 1.

*Ticks Per Bar* — The number of clock ticks per ramp.

*Send Start/Stop/Continue Events* — Sends the appropriate events when the framebar starts or stops.

# 43 MOUSE CHOP

## 43.1 DESCRIPTION

This CHOP outputs X and Y screen values for the mouse device.

The Mouse X and Y positions are output through the channels named in the Position X and Y parameters, when the *Active* button is enabled.

The range of the mouse values are -1 to +1 in X, and -0.8 to +0.8 in Y. 0, 0 is the center of the screen.

The Mouse and Keyboard CHOPs are often connected to the Position and Active inputs respectively of the Record CHOP to enable the recording of channels generated by the Keyboard and Mouse.

## 43.2 PARAMETERS – CONTROL PAGE

### ACTIVE

While *On*, the mouse movement will be output from the CHOP, and the CHOP will cook every frame. Otherwise the CHOP will not cook and the current mouse X or Y values will not be output. If set to *While Playing*, output will only occur while Houdini's playbar is playing.

### POSITION X

The name of the channel that records the horizontal movement of the mouse.

### POSITION Y

The name of the channel that records the vertical movement of the mouse.

## 43.3 PARAMETERS – CHANNEL PAGE

This sets the Sample Rate and the Extend Conditions. See *Extend Options* p. 303.

## 43.4 STANDARD OPTIONS AND LOCAL VARIABLES

- *Units Options* p. 297
- *Extend Options* p. 303
- There are no local variables.

# 44 NOISE CHOP

## 44.1 DESCRIPTION

The Noise CHOP makes an irregular wave that never repeats, with values approximately in the range -1 to +1. It appears to have a period like the Wave CHOP, but it is not exact. It generates both smooth curves and noise that is random each sample. It uses the same math as the Noise Texture Operator.

Optionally, an input can be connected. It is assumed that the input contains 1 to 3 channels representing X, Y and Z coordinates of points in space, and are used to sample anywhere in 3D noise space. One index in the input produces one sample in the output.

## 44.2 PARAMETERS – NOISE PAGE

### TYPE

The noise function used to generate noise. The functions available are:

| | |
|---|---|
| *Sparse* | Produces high quality, continuous noise based on Sparse Convolution. |
| *Hermite* | Quicker than *Sparse*, but produces lower quality noise. |
| *Harmonic Summation* | Sparse noise with the ability to control the frequency step of the harmonics. Takes the longest to compute. |
| *Brownian* | Works like a bug in random flight. With *Num of Integrals* at 2, its acceleration is changed randomly every frame. |
| *Random (White Noise)* | Every sample is random and unrelated to any other sample. It is the same as “white noise” in audio. |

### SEED

Any number, integer or non-integer, which starts the random number generator. Each number gives completely different noise patterns, but with similar characteristics.

### PERIOD

The approximate separation between peaks of a noise cycle. It is expressed in Units. Increasing the period stretches the noise pattern out.

Period is the opposite of frequency. If the period is 2 seconds, the base frequency is 0.5 cycles per second, or 0.5Hz for short. Hz refers to *Hertz*, the electrical and audio engineer of the 19th century, not the car guy.

If the *Type* is set to *Random*, setting this to zero will produce completely random noise. Otherwise, the period should be greater than zero.

### HARMONICS

The number of higher frequency components to layer on top of the base frequency. The higher this number, the bumpier the noise will be (as long as roughness is not set to zero). 0 harmonics give the base shape.

Harmonics with a base frequency of 0.1Hz will by default produce harmonics at 0.2Hz, 0.4Hz, 0.8Hz, etc. (up to the number of harmonics specified by the Harmonics parameter).

### HARMONIC SPREAD

The factor by which the frequency of the harmonics are increased. It is normally 2. A spread of 3 and a base frequency of 0.1Hz will produce harmonics at 0.3Hz, 0.9Hz, 2.7Hz, etc.. This parameter is only valid for the *Harmonic Summation* type.

### ROUGHNESS

Controls the effect of the higher frequency noise. When roughness is zero, all harmonics above the base frequency have no effect. At one, all harmonics are equal in amplitude to the base frequency. When roughness is between one and zero, the amplitude of higher harmonics drops off exponentially from the base frequency.

The default roughness is 0.5. This means the amplitude of the first harmonic is 0.5 of the base frequency, the second is 0.25, the third is 0.125. The harmonics are added to the base to give the final shape. The *Harmonics* parameter and the *Roughness* parameter must both be non-zero to see the harmonic effects.

### EXPONENT

Pushes the noise values toward 0, or +1 and -1. (It raises the value to the power of the exponent.) Exponents greater than one will pull the channel toward zero, and powers less than one will pull peaks towards +1 and -1. It is used to reshape the channels.

### NUM OF INTEGRALS

Defines the number of times to integrate (see the *Area CHOP* p. 309) the Brownian noise. Higher values produce smoother curves with fewer features. Values beyond 4 produce somewhat identical curves. This parameter is only valid for the *Random* noise type.

### AMPLITUDE

Defines the noise value's amplitude (a scale on the values output).

## 44.3 PARAMETERS – TRANSFORM PAGE

### TRANSLATE / ROTATE / SCALE / PIVOT

The *Translate*, *Rotate*, *Scale* and *Pivot* parameters let you sample in a different part of the 3D noise space. Imagine a different noise value for every XYZ point in space. Normally, the Noise CHOP samples the noise space from (0,0,0) along the X-axis in steps of 2/period.

By changing the transform, you are translating, rotating and scaling the line along which the Noise CHOPs samples the noise space. A slight Y-rotation is like walking in a straight path in the mountains, recording your altitude along the way, then re-starting from the same initial location, walking in a slightly different direction. Your altitude starts off being similar but then diverges.

Optionally, an input with one to three channels can be connected as an input to the CHOP. The input is interpreted as X Y and Z coordinates where the noise space is to be sampled. If there are less than two channels, the missing channels are assumed to be 0. This position can be manipulated with the matrix as well.

## 44.4 PARAMETERS – CONSTRAINTS PAGE

### CONSTRAINT

Constraint and its parameters allows the noise curve to start and/or end at selected values. The average value may also be enforced.

### NORMALIZE

Ensures that all noise curves fall between -1 and 1. Applied before the *Amplitude* parameter. Only valid for *Random* and *Harmonic Summation* noise types, since *Hermite* and *Sparse* noise are always normalized. Normalizing random noise occurs between integrations, producing a more controlled curve.

## 44.5 PARAMETERS – CHANNEL PAGE

These parameters specify the format of the channels to be filled with noise, and act the same as the Channel page in the Wave CHOP. See *Channel Name Creation Options* p. 302.

## 44.6 OPTIONAL INPUT: XYZ SAMPLE POINTS

Normally the noise is sampled along the X-axis in an XYZ noise space. You can sample anywhere in XYZ. See *Parameters – Transform Page* p. 395.

## 44.7 STANDARD OPTIONS AND LOCAL VARIABLES


- *Scope and Channel Name Matching Options* p. 296
- *Units Options* p. 297


## 44.8 LOCAL VARIABLES

*Local Variables* p. 306 – $C, $NC, $I

| | |
|---|---|
| $S | The start index of the noise curve. |
| $E | The end index of the noise curve. |

# 45 NULL CHOP

## 45.1 DESCRIPTION

This CHOP is used a place-holder and does not have a function of its own. It is useful for tweaking data (using Edit Data Channels). It can also be used as the final OP in a CHOP network for exporting. If an unchanging OP like this is used to export, as your net changes you do not need to keep switching the Export flag. This also helps when fetching the result of a CHOP network from another network.

## 45.2 PARAMETERS

There are no CHOP-specific parameters – all data is simply passed through.

For Common parameters, see *Common CHOP Parameters* p. 296.

## 45.3 LOCAL VARIABLES

There are no local variables.

# 46 OBJECT CHOP

## 46.1 DESCRIPTION

The Object CHOP compares two objects and returns information on their relative positions and orientations. The information that can be output is:

- Position of one object relative to another
- Rotation of one object relative to another
- Bearing of one object relative to another
- Single Bearing Angle between two objects
- Distance between the origin of two objects
- Inverse Square of the Distance between two objects

The optional two inputs allow you to compare X,Y,Z points in world space with objects or each other. The inputs are expected to have three channels containing XYZ points. These inputs replace the target and/or reference objects. Object and points can be compared with each other, but "Rotation" mode will always return zero.

## 46.2 PARAMETERS – OBJECT PAGE

The default simply gives you one object's position relative to another. You get the XYZ of the origin of the Target Object relative to the origin and rotation of the Reference Object. That is, you get the XYZ of the target object's origin as if you were at the 0,0,0 location (origin) of the Reference object, looking down the Reference Object's Z-axis.

### TARGET OBJECT

The object that is being compared to the position of the reference object.

### NAME

The Target Object can alternatively be expressed as a text string. This can be useful when the object name needs to be a variable – it allows you to type in a name which may include expressions or variables.

### REFERENCE OBJECT

The object that acts as the origin of the comparison.

### NAME

The Reference Object can also alternatively be expressed as a text string.

### COMPUTE

The information to output from the objects, as described below:

| | |
|---|---|
| *Position* | The displacement from the reference object to the target object. |
| *Rotation* | The orientation difference from the reference object to the target object. |
| *Bearing* | The rotation necessary for the reference object to be facing the target object. |
| *Single Bearing Angle* | An angle representing where the target object is relative to the reference object. Zero degrees is directly in front, 90 degrees is beside and 180 degrees is behind. |
| *Distance* | The distance between the two objects. |
| *Inverse Square Distance* | The inverse squared distance between the two objects, useful for modeling electric forces, audio dropoff and gravity. |

### ROTATE ORDER

The rotation order to use for "Rotation" or "Bearing" computation.

### BEARING REFERENCE

Bearing requires a direction to use as a reference base.

### BEARING VECTOR

An arbitrary base direction for the bearing calculation.

### POINT SCOPE

When one of the optional point inputs is connected, this determines which channels represent X, Y and Z.

## 46.3 PARAMETERS – CHANNEL PAGE

Channel names are created automatically.

### START / END

The start and end time of the desired interval of the object path.

***Note:*** When creating rotation channels, the Transform and Object CHOPs will select values which minimize frame-to-frame discontinuity. The graphs will appear continuous and free of 180 degree shifts.

## 46.4 STANDARD OPTIONS AND LOCAL VARIABLES

- *Scope and Channel Name Matching Options* p. 296
- *Sample Rate Match Options* p. 297
- *Units Options* p. 297
- *Extend Options* p. 303
- There are no local variables.

# 47 OSCILLATOR CHOP

## 47.1 DESCRIPTION

The Oscillator CHOP generates sounds in two ways. It synthesizes tones using a choice of common waveforms, or it repeats a prepared incoming audio clip of any duration.

When it is synthesizing tones from the basic waveforms, it steps through the waveform at a rate that depends on the *Pitch Control* input. By default, a Pitch Control of 0 gives a middle A at 440 Hz; A 1 gives 880 Hz; A -1 gives 220 Hz. Steps of 1 in *Pitch Control* are 1 octave apart. Steps of 1/12 (.08333) are 1 semitone apart.

Up to three input CHOPs can be used.

### PITCH CONTROL

The first input affects the pitch. (It is "logarithmic".) Output channels are generated for each Pitch Control channel.

### RESET PULSE

The second (optional) input contains pulses that restart the oscillator from the beginning of the wave or the Playback Source. 0 in the input means "play the oscillator". 1 means "stop the oscillator and cue it at the start of the waveform or Playback Source".

### PLAYBACK SOURCE

The third (optional) input is a replacement of the waveform Type. It is a sound clip to play at a rate modified by the Pitch Control, and can contain any number of channels. These channels are generated for each Pitch Control channel. The Waveform Type and the Base Frequency parameters are disabled.

If you plug any sound clip into the Oscillator CHOP's Playback Source, and Pitch Control is a constant value of 0 of any duration, it will just repeat the Playback Source. If you feed a Wave CHOP as its Pitch Control, it will raise and lower the speed/pitch of the input.

The Oscillator CHOP can serve as a general motion time-warper. If you put motion channels into the third input, you can control the time warp by feeding different Pitch Control curves. 0 pitch is normal speed, 1 is double speed.

Unlike the Waveform CHOP, this is an iterating CHOP, that is, it steps through the waveform while the pitch changes. To see this effect, feed a Wave CHOP into the Oscillator.

## 47.2 PARAMETERS - WAVEFORM PAGE

### TYPE

The shape of the waveform to repeat, unless overridden by the Playback Source: *Sine* (-1 to 1), *Gaussian* (0 to 1), *Triangle* (-1 to 1), *Ramp* (0 to 1), *Square* (-1 to 1), *Pulse* (0 to 1).

### BASE FREQUENCY

The cycles per second when the Pitch Control is zero.

### UNITS PER OCTAVE

Amount that the Pitch Control needs to increase by to raise the pitch by one octave. The default of 1 means that Pitch Control of 1 raises the pitch by 1 octave. Units per Octave of .08333 means that a Pitch Control of 3 raises the pitch by a factor of 3 x .08333 (three semitones). This is suitable for MIDI note numbers.

### OFFSET

Values output from the CHOP can have an offset added to them.

### AMPLITUDE

Values output from the CHOP can be scaled.

### BIAS

Shape control for Triangle, Gaussian and Square waves. For triangle waves, it moves the peak. For square waves, it alters the width of the peak. Zero means no bias.

### PHASE

A value of .5 is a phase shift of 180 degrees, or one half cycle.

## 47.3 PARAMETERS – CHANNEL PAGE

### SAMPLE RATE

The sample rate of the CHOP.

## 47.4 LOCAL VARIABLES

- No local variables.

# 48 PASS FILTER CHOP

## 48.1 DESCRIPTION

This CHOP filters audio input using one of four different filter types.

### SOME FILTER TERMINOLOGY

A Low pass filter removes the higher frequencies of a sound, while a high pass filter reduces the bass of the sound. A band pass filter is used to extract a frequency range (i.e. extracting a person's voice from background clutter) and a band stop filter is used to cut out a frequency range.

If a certain frequency lies outside the pass band, sounds at that frequency will be reduced in magnitude. The farther outside the pass band the frequency is, the more it will be reduced.

The Cutoff frequency is also known as the "half-power" frequency. A wave at the cutoff frequency will be reduced to half power.

The Rolloff Factor of a filter determines how quickly the drop occurs at its Cutoff frequencies. A low rolloff will produce a gradual filter falloff (more of the sounds outside the frequency range are heard), and a high rolloff will produce a sharp filter falloff.

## 48.2 PARAMETERS

### FILTER

| | |
|---|---|
| *Low Pass Filter* | All frequencies below the High Cutoff are passed through the filter (the "pass band"). |
| *High Pass Filter* | All frequencies above the Low Cutoff are passed through. |
| *Band Pass Filter* | All frequencies between the Low and High Cutoff are passed through. |
| *Band Stop Filter* | All frequencies above the High Cutoff and below the Low Cutoff are passed though. |

### LOW CUTOFF

The frequency (in Hertz) of the lower cutoff. This value is not used by a low pass filter, since it has only an upper cutoff.

### HIGH CUTOFF

The frequency (in Hertz) of the upper cutoff. This value is not used by a high pass filter, since it has only a lower cutoff.

### PASS GAIN (DB)

Defines the gain of passed frequencies, specified in dB (decibels). Every increase of 20dB corresponds to a 10 times power increase (and decreases by 20dB similarly reduces power to 0.1 the original).

### ROLLOFF FACTOR

Defines how the filter drops off at the cutoff frequencies. Low values (less than one) produce more gradual rolloff, and higher values produce sharper filters.

### ALSO FILTER PHASE

Normally, the magnitude of the signal is filtered. You can optionally filter the phase of the signal as well. This takes about twice as long, and normally doesn't produce audible differences. It is included for special cases.

## 48.3 FILTER ANIMATION CHANNELS INPUT

The second input is the Filter Animation Channels, which allows the filter parameters to be changed over the CHOP's interval. See *Common Options of CHOPs > Filter Animation Channels* p. 305.

## 48.4 PARAMETERS – DIGITAL PAGE

### DIGITAL FILTER OVERVIEW

Digital filters use a Discrete Fourier Transform (DFT) to filter the sampled data. A digital filter processes chunks of data, unlike an analog filter, which continuously processes data. The digital filter divides the sample into chunks which are each individually filtered. This causes discontinuities in the filtered sample at each division, which must be removed.

By discarding the extreme ends of a chunk, and then slightly overlapping and blending it with the previous chunk, these discontinuities can be eliminated.

The chunk size of a digital filter affects its characteristics. Smaller chunks allow more control of the filter when filter parameters are animated, but have coarser frequency resolution. At very low frequency resolution, the filter will have a grainy sound.

When filter parameters are animated, they are only evaluated once per chunk. If the chunk is large, then actual steps between chunks may be heard, rather than a smooth transition (assuming the animation was meant to be smooth!).

The default values of this page are good for general use, and normally do not need to be changed.

### FILTER CHUNK

This parameter allows you to select the chunk size, to balance the demands of animating filter parameters with the quality of sound.

### CHUNK OVERLAP

How much of the current chunk is overlapped and blended with the previous chunk. It can range from zero (no overlap and blending) to 0.95 (95% overlapped). For practical use, keep it between 0.05 to 0.3 . Values close to one will produce long cook times, and values too close to zero will not entirely remove the discontinuity.

### CHUNK DISCARD

How much of the chunk to throw away. Since the middle section is most accurate, the end sections are discarded. This parameter can also range from zero (don't discard anything) to 0.95 (keep only the middle 5%). Normally, discard between 0.05 and 0.2 of the chunk. Larger values will produce long cook times.

## 48.5 STANDARD OPTIONS AND LOCAL VARIABLES

- *Scope and Channel Name Matching Options* p. 296
- *Sample Rate Match Options* p. 297
- *Local Variables* p. 306 – $C, $NC, $I

| | |
|---|---|
| $N | The current digital chunk being processed. |

# 49 PARAMETRIC EQ CHOP

## 49.1 DESCRIPTION

This CHOP filters an audio clip using "parametric equalizer" type controls, and then applies other audio effects. The functions, in the order they are applied, are:

| | |
|---|---|
| Parametric Filter | Filters any frequency range of the input, using center frequency, bandwidth, filter gain, pass gain, filter shape and dropoff parameters. |
| Sideband Filter | Filters the input with another audio channel's power spectrum (contained in the second input). |
| Pitch Shift | Shifts the pitch of an audio clip (the first input), maintaining the same duration. |
| Echo Generator | Adds echoes to the audio clip. |

### FILTER ANIMATION CHANNELS

The third input is the *Filter Animation Channels*, which allows the filter parameters to be changed over the CHOP's interval. See *Common CHOP Parameters > Filter Animation Channels* p. 305. For example, to animate the Center Frequency, create a CHOP containing an animated channel named "center", and attach it to the third input.

## 49.2 PARAMETERS – FILTER PAGE

This boosts or reduces the audio in a frequency range. The center frequency, frequency width, filter gain, pass gain, filter shape and dropoff are independently controllable.

### FILTER CHANNELS

Turns the parametric equalizer filter effect on or off (but not the other special functions).

### CENTER FREQUENCY

Defines the center frequency of the filter. This frequency will have a gain equal to that of the *Filter Gain*.

### BANDWIDTH

Defines how many octaves around the center frequency is affected by the filter. When the filter is shaped as a Gaussian or a triangular filter, the filter gain drops off from the center frequency along the Gaussian or triangular curve. At a bandwidth of 1, frequencies half an octave above and below the center frequency will be filtered out according to *Pass Gain*.

### FILTER GAIN

Defines the gain of the filter. The audio in the frequency range of the filter is multiplied by the *Filter Gain*. The audio within the band is multiplied by *Filter Gain*.

### PASS GAIN

Defines the gain of the frequencies outside the filtered region. If zero, frequencies outside the range are not audible. The audio outside the band is multiplied by *Pass Gain*.

### FILTER SHAPE

The shape of the filter can be a box, triangle, a gaussian, or any interpolation between. A value of zero produces a pure box, 0.5 produces a pure triangle, and 1 produces a Gaussian shape. Values between 0 and 0.5 produce a filter that is a combination of a box and a triangle, and so on.

Box filters have abrupt drop-offs. Triangular filters have gradual linear drop-offs, with discontinuities at the corners. Gaussian filters have smooth drop-offs.

### FILTER DROPOFF

The shape of the filter can be further modified by changing its dropoff. At one, the filter shape is typical, dropping about 6dB per octave. Values below 1 cause the filter to dropoff more slowly. Values above 1 cause the filter to dropoff rapidly.

## 49.3 PARAMETERS – SIDEBAND PAGE

### SIDEBAND FILTER

Which method to use when sideband filtering: *Average Power Spectrum*, or *Continuous Power Spectrum*.

### FILTER TYPE

The power spectrum of a sideband filter can be used to enhance frequencies.

| | |
|---|---|
| *Sideband Pass* | Enhances frequencies. |
| *Sideband Stop* | Removes frequencies. |

#### SIDEBAND GAIN

The gain of the sideband filter.

#### BASE GAIN

The base gain of the channel.

#### SIDEBAND EFFECT

The frequency effect of the filter.

## 49.4 PARAMETERS – PITCH PAGE

#### PITCH ADJUST

Turns pitch shifting on or off.

#### OCTAVE SHIFT

The pitch of a sound can be shifted up to half an octave up or down by this function. This is expressed in octaves. Note that the more a pitch is shifted, the more it is distorted. A value of zero does not affect the sound.

#### PITCH CHUNK

The chunk size at which *Octave Shift* resamples. Mainly used to fine tune the sound once the correct pitch is found.

## 49.5 PARAMETERS – ECHO PAGE

#### ECHO

Turns echoes on or off.

#### PRE ECHOES

Specifies the number of echoes that precede the actual sound. (Note that pre echoes are not a naturally occurring phenomenon).

#### PRE ECHO DELAY

Specifies the delay between the pre-echoes (in either seconds, frames or samples, depending on the setting of the *Units Options* p. 297 parameter on the *Common* page).

### PRE ECHO DROPOFF

Going back in time, the pre-echoes reduce in magnitude. Pre-Echo Dropoff is the portion of the last pre-echo's magnitude that the current pre-echo's magnitude is. A value of 1 means that all pre-echoes will be equal in magnitude, while a value of 0.5 will produce pre-echoes of magnitudes 0.5, 0.25, 0.125... of the original sound.

### POST ECHOES / POST ECHO DELAY / POST ECHO DROPOFF

Post echoes are naturally occurring echoes: the echoes occur after the original sound. The parameter descriptions are the same as for pre-echoes.

### REMAINDER

See *Remainder Options* p. 305.

## 49.6 PARAMETERS – DIGITAL PAGE

See the *Digital Filter Overview* p. 404 in the Pass Filter CHOP for a full description of digital filters and the parameters on this page.

## 49.7 STANDARD OPTIONS AND LOCAL VARIABLES


- *Scope and Channel Name Matching Options* p. 296
- *Sample Rate Match Options* p. 297
- *Units Options* p. 297
- *Local Variables* p. 306 – $C, $NC, $I

# 50 PARTICLE CHOP

## 50.1 DESCRIPTION

This CHOP produces translate and rotate channels to move Objects according to the positions of particles in a POP Network. Channels are generated for all particles with an *instance* attribute and which have the *override* state turned on.

## 50.2 PARAMETERS – PARTICLE PAGE

### POP NETWORK / NAME

Particle network containing POP, and the POP to cook.

### METHOD

Specifies whether the CHOP should generate channel values for the current frame (Dynamic) or for all frames at once (Cached).

### EXTRACT SCOPE

Specifies a mask of channels to generate.

### UPDATE

Forces an update of the CHOP if it is in *Static* mode.

## 50.3 PARAMETERS – STANDARD PAGE

### START TIME *(/timestart)*

Time at which simulation starts.

### PREROLL TIME *(/timepreroll)*

At start time, the simulation has already been running this long.

### INITIAL STATE

Geometry to use as initial state of simulation.

### OVERSAMPLING *(/oversample)*

How many times to cook in between frames.

#### MAX # OF PARTICLES */maxparticles*

Controls the maximum number of particles that can exist in the simulation at any given moment. A value of 0 means particles can always be birthed.

#### REMOVE UNUSED POINTS

Removes unused points from input geometry.

## 50.4 PARAMETERS – RBD PAGE

#### REST THRESHOLD */restthreshold*

The minimum relative velocity two RBD particles must have to be considered colliding.

#### CONTACT TOLERANCE */contacttol*

The distance at which RBD particles are treated as being in contact.

#### MAX TIME SPLITS */maxdivisions*

A measure of how accurately RBD collisions should be detected.

#### CONSTRAINT TIME */constrainttime*

Approximate time allowed to satisfy RBD constraints.

#### SOLVER TYPE

Type of solver to be used by the RBD solver.

## 50.5 PARAMETERS – INPUT GEO PAGE

#### OBJECT / SOP

The names of the Objects / SOPs to use as Context Geometry.

#### CONTEXT GEOMETRY

All POP networks can have 'Context Geometry'. These context geometries are set in the parameters of the POPNET, the Particle CHOP, or the POP OP. In the POP OP they can also be set by wiring OPs into the inputs of the POP OP. These context geometries can be referenced by the following POPs: Source POP, Collision POP, SoftBody POP, Attractor POP, Creep POP (All of which have the parameter – *Geometry Source > Use Context Geometry)*. So, instead of hard-coding a specific piece of source geometry or collision geometry, a single POP network can be used

by several POP OPs to generate several different outputs, depending on the geometry input into the POP OP.

## 50.6 PARAMATERS – CHANNEL PAGE

### START / END *(/start /end)*

The start and end time of the desired interval of the object path.

### SAMPLE RATE *(/rate)*

The sample rate of the channels.

### EXTEND LEFT / RIGHT

The left / right extend conditions.

### DEFAULT VALUE *(/defval)*

The default value for extend conditions.

# 51 PHONEME CHOP

## 51.1 DESCRIPTION

This CHOP translates english text into a series of phonetic values. The samples of the output channel represents the phonetic values. The phonemes are initially placed at regular intervals along the length of the channel.

## 51.2 PARAMETERS – PHONEME PAGE

**TEXT** */text*

The english text to be translated.

**FILE** */file*

Additional text to be translated.

**NUMBERS** */numbers*

If *Expand*, then numbers are expanded (e.g. 123 = one hundred and twenty three).

**MONEY** */money*

If Expand then money will be expanded (e.g. $4.00 = four dollars).

**PUNCTUATION** */punc*

If *Expand* then punctuation will be expanded into phonetic symboles (for example, ! = exclamation mark).

**CALCULATE PHONEMES**

Click this button to automatically recreate the phonetic translation of the text.

## 51.3 PARAMETERS – EDIT PAGE

| | |
|---|---|
| *Phoneme* | The phonetic value to add or replace. |
| *Remove* | Remove the currently selected phonemes. |
| *Add* | Add the phoneme specified by the Phoneme menu at the current time. |
| *Replace* | Replace the currently selected phonemes with the phoneme specified by the Phoneme menu. |

## 51.4 PARAMETERS – CHANNEL PAGE

See *Channel Page* p. 302 for a description of these parameters:

- *Start-End Options* p. 302
- *Extend Options* p. 303

The first input is an optional start/end reference. If connected, it will override the parameters in the Channel page.

The output channel contains one handle for each phoneme. Phonemes belonging to the same word are colored identically. Individually selected handles can be moved vertically to change their phoneme value. When a group of handles is selected, their outermost handles affect the scale, while any of the inner handles affect their horizontal placement.

## 51.5 PHONEME VALUES

```
0 sil silence   9 uh  fUll    18 b   Back    27 s   Sue     36 w   Wear
1 iy  bEEt     10 uw  fOOl    19 t   Time    28 z   Zoo     37 y   Young
2 ih  bIt      11 er  mURdER  20 d   Dime    29 sh  leaSH   38 r   Rate
3 ey  gAte     12 ax  About   21 k   Coat    30 zh  leiSure 39 ch  CHar
4 eh  gEt      13 ah  bUt     22 g   Goat    31 h   How     40 j   Jar
5 ae  fAt      14 ay  hIde    23 f   Fault   32 m   suM     41 wh  WHere
6 aa  fAther   15 aw  hOW     24 v   Vault   33 n   suN
7 ao  lAWn     16 oy  tOY     25 th  eTHer   34 ng  suNG
8 ow  lOne     17 p   Pack    26 dh  eiTHer  35 l   Laugh
```

## 51.6 LOCAL VARIABLES

None.

# 52 PIPE IN CHOP

## 52.1 DESCRIPTION

***Note:*** This CHOP only works on IRIX; it is not tested under Linux or NT.

This CHOP is a mechanism that allows users to input from custom devices into a CHOP, without needing the Houdini Developers' Kit or knowledge of Houdini internals. It is implemented either as a "named pipe" or "FIFO" (first in, first out) or a network connection (for remote connections).

To make it work, you need a program that gets values from the input device, and outputs them into a file specified in Filename or a Network port. The values are output in a format that is specified in *Programming the Pipe In CHOP* p. 416.

The user's custom program writes formatted messages into the file or port, and the Pipe In CHOP reads it, creating the channels specified in the messages.

One device that is implemented in this way is the Puppetworks device, and its driver is: *$HFS/bin/puppetworks* .

### NETWORKING INTO HOUDINI

You can receive network data from another server (e.g. from a Houdini Pipe Out CHOP running remotely). A connection must be established between the server and the Pipe In CHOP before data is sent. Do this by changing the Source from Pipe to Network). You must supply the Server Address and Port from which to recieve incoming data to a channel. The server should be listening for connections on the port that this CHOP is using.

To setup a link between two Houdini processes, one process should have a Pipe Out CHOP active (thus listening for connections) on an arbitrary port (i.e. port #5010). The second Houdini uses a Pipe In CHOP with the network address set to the name of the machine that the first Houdini process is running on. The network port should be set to the same port as the server (in this case, 5010). The port number can be any number between 5000 and 10000, as long as it is consistent between the Pipe In and Pipe Out CHOPs. For more than one connection, use distinct port numbers. Pipe In/Out CHOPs with matching port numbers on different machines should automatically sense one another.

## 52.2 PARAMETERS

### SOURCE

Data can be piped in through a UNIX pipe or a Network port.

### FILENAME

The file that the device data will be read from. The file must not be a regular file. It must be a "named pipe" or "FIFO". In UNIX, see "mknod".

### SERVER ADDRESS

The network address of the server computer. This address is a standard WWW address, such as 'foo' or 'foo.bar.com'.

### SERVER PORT

The network port of the server. The port is a number between 5000 and 10000, which both the server and the client use to connect with.

### ACTIVE

While active, the CHOP receives information from the pipe or server. When off, no updating occurs. Data sent by a server is lost, but a pipe will store the data until active is turned on again. If in Network mode, turning this parameter on initiates a connection, and turning it off breaks the connection.

### RESET CHANNELS

Clears all channels and data.

## 52.3 PROGRAMMING THE PIPE IN CHOP

The Pipe In CHOP is designed to allow you to program an interface to CHOPs that do not require a Houdini Developers Kit.

The Pipe In CHOP allows Houdini to read information in a special formats described here from a FIFO (First In, First Out) file and create CHOP channels. As long as the correct formats are used, Houdini will be able to get data from any source including devices such as a joystick, microphone or PuppetWorks' hardware.

The Pipe In CHOP has two parameters and parses three command formats. The first parameter specifies which file to read from and the second toggles between active (reading) and inactive (discards data) modes. The formats supported allow the following functionality: upload the most current sample data, upload all sample data and create a full waveform, and upload the channel names.

### USING THE PIPE IN CHOP

In order to read information with this CHOP, a separate application has to be developed to write to a FIFO in the proper format. The following sections describe and give sample code (written in C) for opening a file and writing values that will be accepted by the Pipe In CHOP.

### OPENING A FIFO FILE

To create a file that the Pipe In CHOP will read from, use the following code to open a new FIFO called FIFO_NAME. Define FIFO_NAME to be the desired file name to create. The output stream that is used for writing will block until a reader is connected to the file.

```
#include    <sys/types.h>
#include    <sys/stat.h>
#include    <stdio.h>
mode_t       prev_mask;
FILE        *output = 0;
/* Create a new fifo */
prev_mask = umask(0);
mkfifo (FIFO_NAME, 0666);
umask(prev_mask);
/* Open the file for writing */
fprintf(stderr,
"Awaiting reader of FIFO %s\n\r", FIFO_NAME);
output = fopen(FIFO_NAME, "wb");
```

## WRITING TO THE FILE

The Pipe In CHOP reads information from the FIFO in four-byte chunks of data called tokens, so it is necessary to write to the file in the same format. There is also a method in place for sending an escape character in case a reset is required in the middle of parsing a command.

The reset character is an integer value of 170. When this byte is received, followed by a byte with a value of zero, the current parsing is reset. In order to send a value of 170 without the command being reset, two bytes with a 170 value must be sent consecutively so that the Pipe In CHOP will disregard the first value.

```
#define ESCAPE_CHARACTER        170
void
write_byte(char b, FILE *output)
{
  /* Prepend escape character */
  if (b == ESCAPE_CHARACTER)
    fputc(ESCAPE_CHARACTER, output);
    fputc(b, output);
}
void
write_values(const char *p, int size, FILE *output)
{
  int          i;
  for(i=0; i<size; i++)
    write_byte(p[i], output);
}
void
send_reset(FILE *output)
{
  fputc(ESCAPE_CHARACTER, output);
  fputc(0, output);
  fputc(ESCAPE_CHARACTER, output);
  fputc(0, output);
  fflush(output);
}
```

Using these functions, any four bytes of data (e.g. 0001, 3.141, Chan, ...) can be sent to the file. To send a reset signal, two escape sequences are written just to be sure.

It is a good idea to send a reset signal before the command and flush the FIFO at the end of a command.

```
int         token;
float       sample;
send_reset(output);
/* Send a command here */
write_values((char *)&token, sizeof(token), output);
write_values((char *)&sample, sizeof(sample), output);
fflush(output);
```

### COMMAND TYPE #1 – CURRENT VALUES

The first type of command that the Pipe In CHOP will read is used to get the most recent channel data. It has a default sample rate of 30 samples per second, CHOP length of 1, and a start position of 0. This command allows the number of channels to be set and the samples associated with those channels to be read.

| | |
|---|---|
| *(int) 1* | Command Type. |
| *(int)* | Number of Channels. |
| *(float)* | Sample Data, one sample for each channel. |

### COMMAND TYPE #2 – UPLOAD

The second type of command that can be parsed is used to upload a full set of samples to create a waveform. The sample rate, track length, start position, number of channels, and samples for each channel to fill the track length must be provided. The channel samples must be interleaved so that all the channels for one index are filled before moving forward to the next index.

| | |
|---|---|
| *(int) 2* | Command Type. |
| *(int)* | Track Length. |
| *(float)* | Sample Rate. |
| *(float)* | Start Index. |
| *(int)* | Number of Channels. |
| *(float)* | Sample Data, one sample for each channel for each index. |

### COMMAND TYPE #3 – CHANNEL NAMES

This command allows the channels to be assigned names before they are created. Since the names are usually strings, it is important to remember to write them as four-byte tokens (padded with zeroes at the end if necessary) so they will be parsed correctly. This format requires the number of names to be set, followed by the name length (in number of four-byte tokens) and name data for each channel.

| | |
|---|---|
| *(int) 3* | Command Type |
| *(int)* | Number of Names |
| *(int)* | Name Length, one per name |
| *(char*)* | Name Values, four-byte tokens |

## WRITING A COMMAND

Using command type 1 as an example, the following function could be used to send a command to the FIFO which will be read by the Pipe In CHOP.

```
void
send_current_values(FILE *output, int num, float *samples)
{
  int         token;
  int         j;
  send_reset(output);             /* just to be safe */
  /* Command Type */
  token = 1;
  write_values((char *)&token, sizeof(token), output);
  /* Number of Channels */
  token = num;
  write_values((char *)&token, sizeof(token), output);
  /* Sample Values */
  for(j=0; j<num; j++)
    write_values((char *)&samples[j],
                       sizeof(samples[j]), output);
    fflush(output);
}
```

## SOURCE CODE EXAMPLE

You can find a compilable example of a Pipe In application in:

```
$HH/public/PPD.tar.Z
```

# 53 PIPE OUT CHOP

## 53.1 DESCRIPTION

This CHOP can be used to transmit data out of Houdini to other processes. It can transmit data to a process on the same machine using a UNIX pipe, or it can transmit data to a process running on a remote machine using a network connection. If the other process is another Houdini process, a Pipe In CHOP in that process can be used to receive the data.

For more information on how to connect two Houdini processes running on different machines with a Pipe In and Pipe Out CHOP, see the *Pipe In CHOP* p. 415.

## 53.2 PARAMETERS

### DESTINATION

Data can be transmitted through a pipe or a network connection.

### FILENAME

The filename of the pipe to create and send data through.

### NETWORK PORT

The network port to use. Ports between 5000 and 10000 are available for use.

### ACTIVE

When active, data is sent.

### SEND CURRENT SAMPLE ONLY

If on, only send the current sample. If off, all data between this frame and the last frame is sent.

### SINGLE SAMPLE

In single sample mode, this parameter determines which sample to send; the sample at frame 1 or the current sample.

### SEND ALL DATA

Sends all the channel names and their data in one burst.

# 54 PITCH CHOP

## 54.1 DESCRIPTION

This CHOP is like OCR for waveforms. It will attempt to to detect the dominant pitch in a stream of audio, and synthetically output the detected stripped-down fundamental pitch. It thus recreates the input pitch, and discards all the detail.

It also operates in a second mode which allows you to find the power contribution of individual frequency bands (i.e. the output ouput of a spectrum analyser) over time.

This makes it useful for things like audio generation, or for detecting frequencies within audio. For example, it could be used when you're animating a character, and you want your character to automatically raise an eyebrow everytime the pitch of its voice goes above a certain level (just append a Trigger CHOP after the Pitch CHOP).

## 54.2 PARAMETERS – PITCH PAGE

### PITCH MODE

*Single Dominant Frequency*
: Detects the dominant pitch in a stream of audio, and generates a channel containing this frequency. You could use this as a 'pitch rider' to follow and generate the notes that you sing into a microphone (as MIDI, if you set the Pitch Output in the Channel page).

*Multiple Frequencies*
: Generates a channel to record the power contribution within each specified frequency band.

### FREQUENCY RESOLUTION

The size of the window to sample. Higher numbers yield greater accuracy.

### MINIMUM LEVEL

A cuttoff, so that you don't detect the pitch below a certain power threshold (useful for cutting out detection of background noise).

### PITCH RANGE (MULTI ONLY)

Sets the frequency range within which to detect.

### PITCH STEP (MULTI ONLY)

Since audio is always percieved logarithmically, you will usually want this set to *Logarithmic Divisions*. You can use *Linear divisions*, but it will not correlate to anything useful in audio.

### PITCH DIVISIONS (MULTI ONLY)

When in *Multiple Frequency* mode, this is how many frequency bands to divide the *Pitch Range* into. This would correspond to how many bands you have in a Graphic Equalizer. This is how many channels will be output.

### SMOOTH PITCH CHANGES

When there are abrubt transitions between successive pitches, this will attempt to interpolate smoother transitions between them.

## 54.3 PARAMETERS – DETECT PAGE

*The parameters on this page pertain to Single Pitch Mode*.

### HARMONIC DETECTION

There may be two frequencies which the Pitch CHOP is trying to detect, but the one you want isn't getting detected, because the power of its fundamental frequency is less than that of the other frequency. But if you summed the contribution of the undetected frequency's harmonics, it would be choosen instead. Enable this option to take into account the contribution of the harmonics to the detected pitch.

### HARMONIC ERROR

Allows you to specify a fudge factor for how much your harmonics can be off and still be detected.

### HARMONIC FREQUENCY CORRECTION

Once your harmonics are detected, enabling this option will allow you to refind the base frequency which yields a cleaner detection.

### LIMIT BANDWIDTH

Sets the low and high frequency range in which to detect (does not apply to detection of harmonics however).

### SELECT NEIGHBOURING FREQUENCIES

If the Pitch recognizer is uncertain which of two following pitches to choose, this will put 'dog blinders' on the range it will jump. This prevents popping to values which are really out-of-line with a bunch of previous pitch detects.

### NEIGHBOUR BANDWIDTH

How wide the 'dog blinder' should be.

### USE PITCH HINT / PITCH HINT

If the pitch detector is always guessing too low, you can provide a hint for the recognizer to start from by specifying a frequency here.

### HINT BIAS

How much to favour the hint. A value of 1 will always use the hint, 0 will ignore the hint altogether.

### SMOOTH OCTAVE JUMPS

Say your detected pitch is fluctuating between two values – both of the same note, but residing in different octaves. Enabling this parameter will collapse these instances into the same octave.

### MAX OCTAVE JUMP

Outside of which it'll do the octave collapsing.

## 54.4 PARAMETERS – CHANNEL PAGE

### PITCH/LEVEL SUFFIX

Use these to suffix the created channels.

### PITCH OUTPUT

| | |
|---|---|
| Raw Frequency | This should equal the fundamental detected pitch. |
| Relative to Frequency | The Output pitch relative to the Frequency Specified. |
| Octave Shift from Frequency | Generates output appropriate for feeding back into an Oscillator CHOP. |
| MIDI Notes | Generates MIDI note values from the detected pitch. |

### REFERENCE FREQUENCY

If the Pitch Output is set to Relative to Frequency, this will be the frequency it will be relative to.

### PITCH ADJUST

If Pitch Output is set to *MIDI Notes*, this allows you to offset the output MIDI notes.

### VOLUME ADJUST

You can also adjust the MIDI volume valuse here (you will notice the volume will be adjusted to MIDI volume valuse (between 0-127).

### SAMPLE RATE

Specify how fine the sampling of generated MIDI notes should be.

# 55 PULSE CHOP

## 55.1 DESCRIPTION

This CHOP generates pulses at regular intervals of one channel. The amplitude of each pulse can be edited with the CHOP sliders or with handles on the graph.

The Pulse CHOP can be used as triggers to the Copy CHOP, and can represent regularly-timed events.

By default, the pulses are a single sample long, but you can increase the Pulse Width so that the pulses are steps to the next pulse. You can also interpolate the values between pulses, as Linear, Ease In Ease Out, Cubic or other curves.

The pulses can be restricted to a minimum / maximum limit. If the *Limit Type* is *Clamp*, the graph has additional convenient handles at the minimum and maximum for each pulse.

The Pulse CHOP generates a single channel of up to 32 pulses, and you can merge several Pulse CHOPs into a multi-channel CHOP.

The Pulse CHOP uses its optional second input as a start/end reference, so a number of Pulse CHOPs can be stretched to the same interval.

In order to set the value at the last sample, the option, *Last Pulse at Last Sample* is provided. Otherwise, the last pulse is prior to the last sample.

## 55.2 PARAMETERS – PULSE PAGE

### NUMBER OF PULSES

The number of pulses to generate.

### INTERPOLATE

You can interpolate the values between pulses using the following function curves:

| | |
|---|---|
| *Off* | Disables interpolation. |
| *Linear* | Use linear interpolation between samples when the interval is lengthened. Averages all samples near the new sample when the interval is shortened. |
| *Ease in* | Uses an *easein()* function for blending. |
| *Ease out* | Uses an *easeout()* function for blending. |
| *Ease in Ease out* | Uses both *easein()* and *easeout()* functions. |
| *Cubic* | Cubically interpolates between samples, for smoother curves than Linear. This method is not recommended for channels with sharp changes. |

#### PULSE WIDTH

By default, the pulses are a single sample long, but you can increase the *Pulse Width* so that the pulses are steps to the next pulse.

#### LIMIT TYPE

| | |
|---|---|
| *Off* | Pulses are allowed to be any value with no clamping. |
| *Clamp* | Clamps pulse values to the Minimum and Maximum values specified below. |

#### MINIMUM, MAXIMUM

The pulses can be restricted to a Minimum / Maximum limit. If the *Limit Type* is *Clamp*, the graph has additional convenient handles at the minimum and maximum for each pulse.

#### LAST PULSE AT LAST SAMPLE

In order to set the value at the last sample, the option *Last Pulse at Last Sample* is provided. Otherwise, the last pulse is prior to the last sample.

### 55.3 PARAMETERS – 0 / 8 / 16 / 24 PAGES

#### PULSE X

The value of the pulse.

### 55.4 PARAMETERS – CHANNEL PAGE

#### START / END

The start and end times of the desired interval.

#### EXTEND LEFT / EXTEND RIGHT

The left and right extend conditions.

#### DEFAULT VALUE

The default value for extend conditions.

### 55.5 STANDARD OPTIONS AND LOCAL VARIABLES

There are no local variables.

# 56 RECORD CHOP

## 56.1 DESCRIPTION

This CHOP takes the channels coming in the first (Position) input, converts and records them into an internal storage array, and outputs the storage array as the CHOP output. The optional second (Active) input is used to enable and disable the recording.

During each frame that is being recorded, the Record CHOP uses only the values of the Position input at the current frame. The Type determines how the input values are converted. In one case, input values are recorded as-is  or interpreted as a speed.

If *Record* is set to *Auto Range*, and the Active input goes on, and Houdini is playing, then any existing storage array is cleared, and the Position channels are recorded in a new storage array until Active goes off. To produce a single-frame recording, hold down the Ctrl key during Auto Range recording.

The Mouse and Keyboard CHOPs are often attached to the Position and Active inputs respectively of the Record CHOP to perform the recording of channels from mouse movements, enabled by pressing a keyboard key (see the *Keyboard CHOP* p. 367).

## 56.2 PARAMETERS – CONTROL PAGE

### RECORD

When and how much to record:

| | |
|---|---|
| *Off* | Leaves the output unmodified. |
| *On* | Always records when playing forward. When the active input is in the off state the input is maintained at its current values. |
| *Add* | Adds (offsets) to an existing set of channels previously recorded in the Record CHOP. The channels are left unmodified while the Active input is in the off state. |
| *Auto Range* | Creates an interval based on Active on/off. |
| *Single Sample* | Creates a one-sample channel containing only current values. |

### RECORD INPUT

Determines whether record should sample the time slice or a single frame. You would generally want to use *Current Time Slice*, for audio, as all frames will be evaluated.

If the interval is set to be the *Current Frame*, it will always cook (only look at) the current frame (things downstream still cook regardless of this setting however). Thus, it should generally not be used for audio, but rather fro things like device input, because it interpolates the values between the captured frames.

### INTERPOLATION

Determines how to compute missed input samples using interpolation. Using *Hold Previous Value* does just that; *Linear* and *Cubic* Interpolation will create a mathematical blend of values in a linear (straight line between values in time), or Cubic (smooth round-off curve between beginning and ending values).

### TYPE

The interpretation of the *Position* input:

| | |
|---|---|
| *Raw* | Output values set to the input values as-is. |
| *Position Offset* | The changes in the input (times Gain A/B) are added to the output values. The output only changes when Active is on. |
| *Speed* | Input changes (times Gain A/B) are treated as a speed change. When Active is off, speed is 0. Output is: value_at_previous_frame + speed*timestep. |
| *Speed & Hold* | Like *Speed* except speed is held when Active is off. |

### RECORD OUTPUT

Record can output single frames, a time slice, or the full animation range.

## 56.3 PARAMETERS – INITIAL PAGE

*The parameters on this page are used to initialize the output of the* CHOP *to specific values.*

### POSITION OFFSET

Initial values to use for position (output of CHOP). You can put one value per channel. Values are repeated if not enough are provided.

### SPEED

Initial values to use for speed (for Speed Types). You can put one value per channel. Values are repeated if not enough are provided.

### INITIALIZE

Command menu to set the internal state to be the initial values above. Select an entry to cause the output value to be set to *Position Offset* or *Speed*.

### READ

Command menu to read the current internal state into the initial values above. Select an entry to cause *Position Output* and/or *Speed* to be set to the current values.

### RESET CHANNELS

The current output is cleared and all subsequent channels will commence single sample lengths.

### RESYNC TIME

If you lock and unlock a Record CHOP, and you have moved the Playbar while it was locked, it'll cause a jump in values because it overwrites your recorded channel data with interpolated data. If this option is enabled, the "last time" will be set to your current frame, ensuring that the recorded data is used instead interpolated data.

## 56.4 PARAMETERS – LIMIT PAGE

*Two sets of limits on the output channels are permitted. Often one is for X (or the first input channel), the other for Y (or the second input channel). If more than two channels are coming in, the A and B limits are repeated.*

### CHANNELS PER LIMIT

Number of consecutive channels to use Limit A before switching to Limit B.

### LIMIT A/B

Type of limiting function for sets A and B.

| | |
|---|---|
| *Off* | Do not apply limits. |
| *Clamp* | Clamp at the minimum/maximum value and hold. |
| *Loop* | At the limit, continue channel at other end of min/max range. |
| *Zig Zag* | At the limit, bounce channel back within min/max range. |

### MINIMUM A/B

The minimum value of the A and B channels.

### MAXIMUM A/B

The maximum value of the A and B channels.

### GAIN A/B

The Position input is multiplied by this gain factor.

## 56.5 STANDARD OPTIONS AND LOCAL VARIABLES

- *Sample Rate Match Options* p. 297
- *Scope and Channel Name Matching Options* p. 296
- *Units Options* p. 297
- There are no local variables.

# 57 RENAME CHOP

## 57.1 DESCRIPTION

This CHOP renames channels. Channels names from the input CHOP are matched using the *From* pattern, and are renamed to the corresponding name in the *To* pattern. Channels that do not match the *From* pattern are not affected.

The channel values and the channel order are not affected, only their names change.

(Note that the Constant CHOP (containing channel values of 0) added to another CHOP in a Math CHOP can also be used to rename channels.)

## 57.2 PARAMETERS

The optional second input CHOP *(Name Reference)* names the first CHOP's channels to be the same as the second CHOP's channels, in the order they appear. In this case, *From* and *To* are ignored.

### FROM

The channel pattern to rename. See *Scope and Channel Name Matching Options* p. 296 in the *Standard Options of CHOPs* section.

### TO

The replacement pattern for the channel names. The channel pattern to rename. See *Scope and Channel Name Matching Options* p. 296 in the *Standard Options of CHOPs* section.

## 57.3 SAMPLE NAMING PATTERNS

The *From* and *To* parameters are any channel name creation patterns like: *chan[1-9:2]* or *t[xyz]* . The *From* parameter also allows matching patterns like *chan** and *ch?* . For example:

*Simple matching assuming an input of chan1 chan2 chan3:*

| From | To | Produces |
|---|---|---|
| chan1 chan2 chan3 | tx ty tz | tx ty tz |
| chan[1-3] | t[xyz] | tx ty tz |
| chan[1-3] | tx ty tz rx | tx ty tz |
| chan[1-2] | tx ty | tx ty chan3 |

*Simple matching assuming an input of tx ty tz:*

| | | |
|---|---|---|
| t[xyz] | [XYZ]trans | Xtrans Ytrans Ztrans |

*Change channels that start with "bob" to channels that start with "carol":*

| | | |
|---|---|---|
| bob* | carol* | |

*Replace channels with "hand" in the middle to channels with "foot" in same place:*

| | | |
|---|---|---|
| *hand* | *foot* | |

*Swap words in a name, where the words are separated by _ :*

| | | |
|---|---|---|
| *_* | *(1)_*(0) | a_x changes to x_a |

*Wildcard matching, assuming an input of ch1 ch2 ch3:*

| | | |
|---|---|---|
| ch? | t? | t1 t2 t3 |
| * | op:* | op:ch1 op:ch2 op:ch3 |
| *h* | *han* | chan1 chan2 chan3 |
| *[1-3:2] | foot[11-13:2] | foot11 ch2 foot13 |
| * | chan[3,1,2] | chan3 chan1 chan2 |

## 57.4 STANDARD OPTIONS AND LOCAL VARIABLES

- *Scope and Channel Name Matching Options* p. 296
- *Scope and Channel Name Matching Options* p. 296
- There are no local variables.

# 58 REORDER CHOP

## 58.1 DESCRIPTION

This CHOP reorders the first input CHOP's channels by numeric or alphabetic patterns. Either a channel pattern specifies the new order, or a number sequence specifies the new order.

If the second input, the *Order Reference* is present, the Numeric Pattern and Character Pattern are ignored, and the first input CHOP's channels are reordered to match as well as possible the reference CHOP's. In this case, Method is not used.

Channel values are never affected.

## 58.2 PARAMETERS

### METHOD

There are three different reordering methods. You can enter a Numeric Pattern, a Character Pattern, or use an optional second input CHOP as an order reference.

### NUMERIC PATTERN

This reorders the channels by channel number. Normally the index order is 0,1,2,3... etc.. The first channel is at index 0. Standard numeric patterns are allowed such as "0-6:1,2" or "!*:1,3".

### CHARACTER PATTERN

This reorders the channels by channel name. Standard character patterns are allowed such as "ch[XYZ]" or "chan[1-15:2,5]" or "chan? ch*". See *Scope and Channel Name Matching Options* p. 296 in the section, *Standard Options of CHOPs*.

### REMAINING POSITION

Channels that do not match are called "remaining" and can also be ordered: they can be placed at the *At Beginning* or *At Ending* (in reference to the position of the matched channels).

### REMAINING ORDER

The channels that did not match can have the *Same as Input* order, or can be sorted *AlphaNumeric*-ally.

## 58.3 EXAMPLES

All examples assume the Remaining Position is *At Ending* and the Remaining Order is *Same as Input*.

| Input | Numeric Pattern | Result |
|---|---|---|
| A B C D | 3 2 1 0 | D C B A |
| C B D A | 3 1 | A B C D |
| A B C D E[1-5:2] | | A C E B D |
| A1 A2 B1 B2 | *1 *2 | A1 B1 A2 B2 |
| c4 c2 c3 c1 | c[1-4] | c1 c2 c3 c4 |

## 58.4 STANDARD OPTIONS AND LOCAL VARIABLES

- *Scope and Channel Name Matching Options* p. 296
- *Scope and Channel Name Matching Options* p. 296
- There are no local variables.

# 59 RESAMPLE CHOP

## 59.1 DESCRIPTION

This CHOP resamples an input's channels to a new rate and/or start/end interval. In all cases, the entire input interval is resampled to match the output interval.

## 59.2 PARAMETERS – RESAMPLE PAGE

### METHOD

The resample method to apply to the channels:

*Same Rate, New Interval* Stretches or compresses the channels like the Stretch CHOP.

*New Rate, Same Time Range*
Changes the sample rate without changing the time-length of the CHOP.

*New Rate, Same Index Range*
Changes the sample rate without changing the number of samples in the CHOP.

*New Rate, New Interval* Changes both the sample rate and stretches/compresses the CHOP.

### SAMPLE RATE

The new sample rate.

### UNIT VALUES

Determines how the Start/End parameters are interpreted.

*Absolute* The value is the new start/end position.

*Relative to Start/End* The value is a shift from the old start/end position.

### START / END

The CHOP's new start and end positions.

### INTERPOLATION

The interpolation method to use when resampling:

| | |
|---|---|
| *No Interpolation* | Use the value of the nearest sample. |
| *Linear* | Use linear interpolation between samples when the output has more samples. Averages all samples near the new sample when the output has fewer samples. |
| *Cubic* | Cubically interpolates between samples, for smoother curves than Linear. This method is not recommended for channels with sharp changes. |
| *Pulse Preserve* | A linear interpolation that recognizes single sample pulses and preserves their height and one-sample width. A pulse is a non-zero value proceeded and followed by zero-value samples. |

### CONSTANT AREA

If enabled, keeps the area under the channel constant by scaling the values of the channel.

### CORRECT FOR CYCLES

If enabled, compensates for cyclic channels (such as angles) by always choosing the shortest step between samples, like 360 to 0 for angles.

### CYCLE LENGTH

The length of the cycle. 360 is typical for angles.

## 59.3 STANDARD OPTIONS AND LOCAL VARIABLES

- *Interpolation Method Options* p. 304
- *Start-End Options* p. 302
- There are no local variables.

# 60 SEQUENCE CHOP

## 60.1 DESCRIPTION

This CHOP takes all its inputs and appends one CHOP after another. It is expected they all have the same channels.

The end section of the first CHOP is overlapped with the start section of the second CHOP, and so on for the rest of the input CHOPs. The second input is shifted to line up with the end of the first.

Blending allows you to splice channels together by slowly phasing out one CHOP while phasing into the next, or by inserting interpolation curves between the channels of the adjacent CHOPs.

Quaternion Blend blends rotation triplets (rx ry rz) together using the shortest rotation arc. Rotation triplets are identified by "quaternion" attributes, which are set in the *Attribute CHOP* p. 311.

Translation Blending blends translation channels together by slowly changing from the final velocity of the previous channel to the initial velocity of the next. The next channel may be shifted up or down. If this is undesirable, use cubic blending instead (in the *Shape* menu). Translation Blending is done on channel triplets that represent translations or positions (*tx *ty *tz).

## 60.2 PARAMETERS – BLEND PAGE

### METHOD

The blend method to produce a seamless sequence:

| | |
|---|---|
| *Preserve Length* | The total length of the CHOPs is constant. |
| *Overlap Sequences* | Overlaps the current CHOP with the prior CHOP. Better for audio. |
| *Insert Blend Region* | Inserts a region between the CHOPs where the blending is done. |

### SHAPE

The blend interpolation shape to use. See *Shape* p. 330 in the *Cycle CHOP* p. 329.

### REGION

The size of the blend region.

#### BIAS

Which segment to favour when blending:
the previous (-1), the next (+1) or neither (0).

#### MATCH BY

Match channels between inputs by index or by name.

## 60.3 PARAMETERS – SCOPE PAGE

#### STEP

If set to 1, the next segment will be shifted up or down so that it begins where the last segment ended.

#### STEP SCOPE

The names of channels that use *Step*.

#### BLEND SCOPE

The names of the channels that should be blended. Other channels will not be blended.

#### TRANSLATE BLEND

The names of channels that will be translation-blended. Each string field contains a list of its component channels, such as **tx*, **ty* and **tz* .

## 60.4 PARAMETERS – ROTATE PAGE

#### QUATERNION BLEND

Use quaternion blending on rotation channels.

#### SHORTEST PATH ROTATION BLENDING

If enabled, compensate for cyclic channels (such as angles) by always choosing the shortest route between samples when blending.

#### ROTATION SCOPE

Defines which channels are rotations for Shortest Path Rotation Blending.

#### CYCLE LENGTH

The length of the cycle for Shortest Path Rotation Blending.

## 60.5 STANDARD OPTIONS AND LOCAL VARIABLES

- *Sample Rate Match Options* p. 297
- *Units Options* p. 297
- *Match By Options* p. 304
- *Overlap Options* *
- There are no local variables.

* All of the input CHOPs will be overlapped with the same parameters. If you want the overlapping to be different for each pair of inputs, you will have to use two or more Sequence CHOPs. The default overlap parameters gives no overlapping, just simple appending.

# 61 SHIFT CHOP

## 61.1 DESCRIPTION

This time-shifts a CHOP, changing the start and end of the CHOP's interval. However, the contents of the channels remain the same.

Each channel can be shifted a different amount by using the $C variable in the Scroll parameter or the Start/End parameters.

An optional second input, the *Start/End Reference*, is used to align the first input CHOP relative to another reference CHOP. This outputs the channels of the first CHOP only, and the shifts are based on the interval of the second CHOP. It is useful for making several CHOPs match up to the same time.

## 61.2 PARAMETERS

### REFERENCE

The start or the end of the channels can be used as the reference position. The channels are shifted by altering the reference position.

*Start Position* Uses the start of the CHOP to do the aligning.

*End Position* Uses the end of the CHOP to do the aligning.

### UNIT VALUES

Determines how the Start and End parameters are to be interpreted:

*Absolute* The Start/End parameters are the actual start/end position. It is used, for example, to shift the CHOP to start at 0 seconds.

*Relative to Start/End* The Start/End parameters are a shift relative to the input CHOP's start/end position.

*Use Current Frame* Ignore the Start/End parameters and use the current frame as the new start position.

### START / END

The start / end of the interval, absolute or relative to the input CHOP.

### SCROLL OFFSET

Without changing the length of the CHOP, you can scroll the channels within its range, and scroll each channel a different amount. By using $C in the parameter, you can make the scroll amount dependent on the channel index.

## 61.3 STANDARD OPTIONS AND LOCAL VARIABLES

- *Start-End Options* p. 302
- *Units Options* p. 297
- *Scope and Channel Name Matching Options* p. 296
- *Local Variables* p. 306 – $C, $NC

| | |
|---|---|
| $S | The start of the interval, in samples. |
| $E | The end of the interval, in samples. |
| $L | The length of the interval, in samples. |

# 62 SHUFFLE CHOP

## 62.1 DESCRIPTION

This CHOP reorganizes a list of channels. It is extremely useful for transforming data received by the Geometry and Image CHOPs into channels containing only one row or column. Data can be easily manipulated, then transformed back if needed.

## 62.2 PARAMETERS

### SHUFFLE OPERATION

Chooses the operation "shuffle" performs:

*Swap Channels and Samples*
: Performs a channel transpose operation, by storing all samples at the same index in the same channel. If 25 channels are in the CHOP with a length of 33 samples, 33 channels will be created with a length of 25.

*Sequence Channels By Name*
: Sequence channels together that share the same alphabetic name, in the order of their number. (i.e. chan2, chan3 and chan1 would be sequenced in the order chan1, chan2, chan3).

*Sequence All Channels*
: Sequence all channels in the clip.

*Sequence N Channels*
: Sequence channels in groups of N together. For N=4, channels 0 to 3, 4 to 7, etc. will be sequenced.

*Sequence Every Nth Channel*
: Sequence every Nth channel together. For N=4, channels 0,4,8,.., 1,5,9,..., etc. will be sequenced.

*Split All Samples*
: Split every channel into channels of 1 sample, each containing a different sample from the original channel.

*Split N Samples*
: Split each channel into segments of N samples (specified below).

*Split Every Nth Sample*
: Take every Nth sample from the original to form N new channels.

For all splitting operations, channels will be numbered according to their segment number. (i.e. chan would be split into chan1, chan2, chan3, etc).

### CHANNEL NAMES

The new name(s) of the channels after a "Swap Channels and Samples" operation.

### N VALUE *Inval*

The value of N for Sequence: Every Nth Channel, Sequence N Channels, Split N Samples and Split Every Nth Sample.

## 62.3 LOCAL VARIABLES

*none*.

# 63 SPATIAL CHOP

## 63.1 DESCRIPTION

The SpatialAudio CHOP is the rendering engine for producing 3D audio. It uses the Sound and Microphone objects to define positional sources and microphones, Geometry objects to define obstructions and Acoustic CHOPs to define sound materials and audio filters.

The first input is an optional lookup table for the volume dropoff over distance. This is required if the Distance Volume Loss parameter is set to 'Distance/Volume Lookup Table'. The second input is an optional 'Environment Filter' which describes how the environment affects the frequencies of sound traveling through it (used with the Distance Volume Loss parameter).

## 63.2 PARAMETERS – ENVIRONMENT PAGE

### MICROPHONES

Specifies the microphone objects to use.
One channel is created per microphone.

### SOUND SOURCES

Specifies the sound objects to use.

### METERS PER UNIT *meterperunit*

Length of one world unit, in meters.

### SPEED OF SOUND *soundvel*

The speed of sound of the environment, in meters per second (the default speed is for air).

## 63.3 PARAMATERS – EFFECTS PAGE

### ENABLE DISTANCE DELAY

Enables delays over long distances, as well as the doppler effect (realistic if on).

### DISTANCE VOLUME LOSS

Method for calculating volume loss over distance.

*None* Distance does not diminish sound.

*Realistic Distance Dropoff* Natural volume dropoff (1/d^2)

*Distance/Volume Lookup Table*
Determine from the lookup table connected to the first input.

### 10M VOLUME LOSS */volloss*

How much volume is decreased after traveling 10 meters (0.4=40% decrease).

### VOLUME LOOKUP RANGE */vollookup1-2*

The range of the distance lookup table; (10,100) means that the table describes how the volume behaves from 10-100 meters.

### USE MICROPHONE FILTERS

Enables or disables all microphone filters.

### CHECK FOR OBSTACLES

Turns on the algorithm for obstacle occlusion.

### OBSTACLES

Add geometry objects as an obstacles. The sound material for these object must be defined. The geometry detail should be low.

### COLLISION DETECTION

Specifies the collision detection algorithm:

*Object Bounding Box* Uses the bounding box of the entire object as the collision object (very fast).

*Primitive Bounding Box* Uses a bounding box per primitive. Good for objects that consist of separate geometries.

*Object Geometry* Uses the actual geometry as the collision object (slowest & most accurate).

### OBSTACLE SOFTNESS */obstsoft*

If an object is between a microphone and a sound source, this parameter determines how abruptly the cutoff is (0 = abrupt, 1 = smooth).

## 63.4 PARAMETERS – ECHO PAGE

### ECHO METHOD

Computes static or dynamic environmental echoes.

### NUMBER OF ECHOES */numecho*

The number of echoes to compute.

### ECHO DELAY */echodelay*

The time between echoes.

### ECHO VOLUME */echovol*

Adjusts the volume of all echoes.

### DYNAMIC ECHO EFFECT */dyneffect*

Percentage longer that echoes take to arrive than the initial sound (.5 = 50% longer)

## 63.5 PARAMETERS – CHANNEL PAGE

### COMPUTE FULL ANIMATION RANGE

Compute using the full animation range, otherwise use the Start / End parameters.

### START / END */start /end*

The range to use if the above parameter is not enabled.

### PREROLL */preroll*

The amount computed before the start of the interval.

### OBJECT SAMPLE RATE */objsample*

The rate that object are sampled at.

### AUDIO SAMPLE RATE */audiosample*

The sample rate of the audio produced.

# 64 SPECTRUM CHOP

## 64.1 DESCRIPTION

This CHOP calculates the frequency spectrum of the input channels, or a portion of the channels. The spectrum can be manipulated and then converted back to get a filtered signal.

When converting a signal to its spectrum, two channels are created from the one containing the signal. One channel contains the magnitude of the frequency components, and the other contains the phase. The channels are named <*channel name*><*suffix*>, where <*suffix*> is the magnitude or phase suffix.

In order to convert back to a signal, both channels are required. The suffixes should be the same as those used in the previous spectrum CHOP.

## 64.2 PARAMETERS

### CONVERT

Determines whether to calculate the frequency spectrum from a signal, or reconstruct a signal from a frequency spectrum.

### CONVERT ENTIRE CHANNEL

If on, the spectrum is computed for the entire channel. Otherwise, segments of the channel can be analyzed.

### CONVERSION START / END */start /end*

Determines the starting/ending points of the segment to be analyzed.

### MAGNITUDE SUFFIX

The string appended to the channel name when converting to a spectrum that identifies the channel as containing magnitudes.

### PHASE SUFFIX

Similar to Magnitude Suffix, but for phase channels.

## 64.3 LOCAL VARIABLES

*none.*

# 65 SLOPE CHOP

## 65.1 DESCRIPTION

This CHOP calculates the slope (or derivative) of the input channels.

If the input CHOP represents position, the slope can be interpreted as speed. By default, the Slope CHOP converts position to speed.

In mathematical terms, the slope is the first derivative of the channel curve. The second and third derivatives can also be calculated. The second derivative can be interpreted as acceleration (and the third derivative could be interpreted as the rate of change in acceleration).

This CHOP can be used in conjunction with the Area CHOP to manipulate speed or acceleration directly. You can calculate the speed or acceleration of a moving object with a Slope CHOP and manipulate it with other CHOPs. Then you can convert the new speed or acceleration curve back to position data with an Area CHOP. You may need to adjust the starting position, since the Slope CHOP removes this information. This can be done with the use of the Constant parameters in the Area CHOP.

The *Units* option causes the output to be expressed as a change in value per sample, value per frame, or value per second.

## 65.2 PARAMETERS

### TYPE

The type of slope to calculate.

*Slope* Calculates the slope of the channels.

*Acceleration* Calculates the acceleration of the channels.

*Acceleration of Slope* Calculates the acceleration of the slopes of the channels.

### METHOD

The sample pairs used to calculate the slope.

*Use Previous And Current Sample*
The current sample and the sample before it are used. This is the only method applicable in realtime applications.

*Use Current And Next Sample*
The current and the next samples are used.

*Use Previous And Next Sample*

The previous and the next samples are used, producing slightly more continuous slopes than the other methods.

The slope at a sample is calculated by looking at the values of samples just before, at, and after it. For the *Slope Type*, it needs two values, and the options are given to use three combinations, the default being to use the value at the sample and the one before. (before – current) is divided by the time between the samples (determined by the sampling rate). The slope placed in the channel is expressed in the Units of the CHOP.

For the type *Acceleration*, it uses three consecutive samples around the sample it is computing. The most accurate acceleration is obtained with *Use Previous and Next Sample*.

## 65.3 STANDARD OPTIONS AND LOCAL VARIABLES

- *Units Options* p. 297
- *Scope and Channel Name Matching Options* p. 296
- There are no local variables.

# 66 SPLINE CHOP

## 66.1 DESCRIPTION

This CHOP allows you to edit the channel data by using direct manipulation of cubic or Bezier handles in the graph of the CHOP.

The CHOP first does a best-fit on the existing samples based on the *Tolerance*, the Start-End range and the channel Scope.

The CHOP's entire interval or a sub-interval can be edited. The scope can be used to limit which channels are to be edited. Changing the scope causes any prior hand-edits to be lost.

In the graph, the solid boxes at the ends of the CHOP interval can be moved to shorten the interval to be edited.

The hollow boxes are the handles for editing the interpolation curves. As soon as one of these is moved, the start-end handles or scope cannot be changed without losing the hand-edits. To change the interval or re-edit the curves, turn on *Compute Knots* again. You can shorten the edit interval to one sample and edit a single sample.

## 66.2 PARAMETERS

### TYPE

The type of fitting curves to use: *Bezier* or *Cubic*.

### COMPUTE KNOTS

Turn this on to recompute the knots when you modify the error or the start or end of the sub-range. All previous changes to the knots will be erased. Any change to a knot will turn this parameter off.

### UNIT VALUES

Determines whether the start/end parameters listed below are absolute values or relative to the start and end of the channels.

The next three parameters will not have any effect unless "Compute Knots" is on.

### START / END

The start / end of the sub-range.

### TOLERANCE

The error tolerance when fitting curves to the data. Lower error tolerances produce a more accurate reconstruction of the original data, but may use more curve segments.

It is not yet possible to have multiple intervals.

## 66.3 STANDARD OPTIONS AND LOCAL VARIABLES

- *Scope and Channel Name Matching Options* p. 296
- There are no local variables.

# 67 SPRING CHOP

## 67.1 DESCRIPTION

This CHOP creates vibrations influenced by the input channels, as if a mass was attached to a spring.

It acts as if, for every channel, there is a mass at the end of a spring, affected by a distance from the actual position (the output of the channel at the previous frame) to the desired position (the input channel at the current frame). When the distance (output - input) is zero, there is no force and therefore no movement.

Alternately, when *Input Effect* is force, the input is used as a force on the spring/mass, and the CHOP reacts to this force plus the force of the spring/mass. In this case, the mass would always stabilize at value 0 if the input is a force of 0.

The Damping acts to make the spring system lose energy, so that higher damping makes everything come to rest sooner.

Its behaviour is best understood by feeding it a CHOP that steps from one constant value to another in sequence, then playing with the constants.

## 67.2 PARAMETERS

### SPRING CONSTANT

The strength of the spring. Larger spring constants produce higher frequency oscillations.

### MASS

The mass of the object on the end of the spring. Higher masses will produce lower frequency oscillations, have higher amplitudes, and be more resistant to damping.

### DAMPING CONSTANT

The amount of damping (resistance) applied to the spring action. Higher damping causes oscillations to die off more quickly.

### INPUT EFFECT

Determines whether the input channel(s) represents a position or a force.

### INITIAL CONDITIONS FROM CHANNEL

If on, the initial position and velocity are calculated from the values at the beginning of the channel.

### INITIAL POSITION

The initial position of the mass attached to the spring.

### INITIAL SPEED

The initial velocity of the mass attached to the spring.

## 67.3 STANDARD OPTIONS AND LOCAL VARIABLES

- *Scope and Channel Name Matching Options* p. 296
- *Units Options* p. 297
- *Local Variables* p. 306 – $C, $NC

# 68 STRETCH CHOP

## 68.1 DESCRIPTION

This CHOP preserves the shape of channels and the sampling rate, but resamples the channels into a new interval. All data in the range is stretched or compressed to the new start and end positions.

## 68.2 PARAMETERS – STRETCH PAGE

### INTERPOLATION

The interpolation method to use when resampling.

| | |
|---|---|
| *No Interpolation* | Use the value of the nearest sample. |
| *Linear* | Use linear interpolation between samples when the interval is lengthened. Averages all samples near the new sample when the interval is shortened. |
| *Cubic* | Cubically interpolates between samples, for smoother curves than Linear. This method is not recommended for channels with sharp changes. |
| *Pulse Preserve* | A linear interpolation that recognizes single sample pulses and preserves their height and one sample width. A pulse is a non-zero value preceded and followed by zero-value samples. |

### CONSTANT AREA

If enabled, keeps the area under the channel constant by scaling the values of the channel.

### UNIT VALUES

Determines how Start/End parameters are interpreted:

| | |
|---|---|
| *Absolute* | The value is the new start/end position. |
| *Relative to Start/End* | The value is a shift from the old start/end position. |

### START / END

The CHOP's new start / end position.

### LENGTH SCALE

Scales the length of the channel after determining the start/end interval. Good for doubling or halving the length.

### REVERSE INTERVAL

Reverses the channel so that it plays backwards.

## 68.3 STANDARD OPTIONS AND LOCAL VARIABLES

- *Interpolation Method Options* p. 304
- *Start-End Options* p. 302
- *Units Options* p. 297
- *Local Variables* p. 306 – $C, $NC.

# 69 SUB-NETWORK CHOP

## 69.1 DESCRIPTION

The Subnetwork CHOP is a way of creating a macro to represent a collection of CHOPs as a single CHOP in the Network Editor. The Subnetwork CHOP can contain an entire CHOP Network within it, stream-lining and simplifying the CHOP network both visually and conceptually.

Selecting *Edit Sub-Network...* from the OP's pop-up menu presents you with a new Network Editor with four sub-network inputs. These four inputs are connected directly to the four inputs on the Sub-net OP in your original network. Proceed by attaching OPs as required to these four sub-network inputs. The display OP will be wired back to the output connector of the Sub-net OP in your original OP network. To get back to the original OP network, go up a level (type U).

Please refer to the *Sub-Networks* p. 190 in the *Interface* section for a complete discussion and an example of how to use sub-networks.

***Tip:*** Select several Operators that you want to make into a sub-network, and select *Collapse Selected* from the *Op's pop-up* menu to automatically create a sub-network out of them. You will see the selected Operators replaced by a single Subnetwork CHOP, and it will be properly rewired to contain the previously selected CHOPs.

## 69.2 PARAMETERS

### INPUT #1 - #4 LABEL

These labels are displayed when you click with on one of the Subnetwork CHOP's inputs. This is useful for remembering what the inputs are used for in your Sub-network.

### SUBNET CHOP

This menu designates exactly one CHOP within the subnet as the default output.

## 69.3 STANDARD OPTIONS AND LOCAL VARIABLES

- *Sample Rate Match Options* p. 297

# 70 SWITCH CHOP

## 70.1 DESCRIPTION

The Switch CHOP allows you to control the flow of channels through a CHOPnet. It selects one of the input CHOPs by index and copies it exactly. This is useful for selecting one of several "gestures" or actions. Only one input CHOP can be selected at a time.

Inputs are indexed starting at 0, so that the first input is 0, the second is 1, and so on.

If the *First Input Is Index* parameter is enabled, the first input is used as the "switch" and the remaining inputs will be selection choices. In this case, the second input will be indexed as 0, the third as 1, the fourth as 2, and so on (the Switch input CHOP is not indexed). The index value is determined by evaluating the first channel in the first input at the current frame. Only integer indices are used; fractional indices will be rounded down to the closest integer.

If the index is less than 0, then the index will be interpreted to be 0. If the index is greater than the number of input CHOPs, the last input CHOP is selected.

## 70.2 PARAMETERS – SWITCH PAGE

First Input Is Index

If enabled, it will use the first input CHOP's first channel to determine which input to select.

**INDEX** */index*

The input index to use if *First Input Is Index* is off.

## 70.3 LOCAL VARIABLES

None.

# 71 TRANSFORM CHOP

## 71.1 DESCRIPTION

This CHOP takes translate, rotate, and/or scale channels and transforms them using the transform parameters in the CHOP. It can be used to:

- Change the position and orientation of an object.
- Convert a set of transform channels with a certain transform order into an equivalent set of channels with a different transform order.
- Change the direction, starting point and scale of motion capture data.

At present, only one transform format exists: the transform defined by 12 channels (tx ty tz, rx ry rz, sx sy sz, px py pz) and a transform order.

The Transform CHOP expects the input channels to have names that end in the above names. Examples are:

```
geo1:tx geo1:ty geo1:tz geo1:rx ...
headtx heaty headtz headrx
tx ty tz rx ...  (what you would get from a Fetch CHOP)
```

The above defines a transformation matrix. This is multiplied by the transform matrix defined on the *Transform* page, and output as translate/rotate/etc. channels.

To affect only one set of channels coming into the CHOP, use the *Scope* parameter.

## 71.2 PARAMETERS – TRANSFORM PAGE

This page defines the matrix used to transform the channels. The parameters used to create this matrix are Transform Order, Rotate Order, and the Translate, Rotate, Scale and Pivot parameters.

### OUTPUT CHANNELS

Currently there is only one output type, the transformed input channels.

### MATRIX OPERATION

The input channels can be pre or post multiplied by the transformation.

### USE ROTATION HINT

These parameters allow you to specify approximate starting values for the rotation channels produced. This allows you to change the rotation channel solution to a specific starting point (e.g. for camera output control).

## 71.3 PARAMETERS – INPUT PAGE

This page defines what the incoming channels' transform order is assumed to be. Using the incoming channels and the transform order here, a matrix for the incoming channels in built. It is  then multiplied by the transformation matrix defined by the Transform page and output.

Any missing translation, rotation or scale channels will default to zero (or one in the case of scale).

### TRANSFORM ORDER

The transform order of the input channels.

### ROTATE ORDER

The rotation order of the input rotations.

***Note:*** When creating rotation channels, the Transform and Object CHOPs will select values which minimize frame-to-frame discontinuities. The graphs will appear continuous and free of 180±≥∞ degree shifts. The Transform CHOP can be used to smooth out rotation channels derived from outside sources, such as the *origin()* function.

## 71.4 STANDARD OPTIONS AND LOCAL VARIABLES

- *Scope and Channel Name Matching Options* p. 296
- *Local Variables* p. 306 – $I (The current index).

# 72 TRIGGER CHOP

## 72.1 DESCRIPTION

This CHOP adds an audio-style attack/decay/sustain/release (ADSR) envelope to all trigger points in the input channels. A trigger point occurs whenever the first input's channel increases across the trigger threshold value.

The envelope consists of six major sections: delay, attack, peak, decay, sustain and release.

From the time the threshold is reached and while the channel's value is above the release threshold, the envelope is in its sustain phase during which it will delay, attack, peak-hold, decay and then maintain its sustain value. Then the envelope will decay to 0.

The peak and sustain levels can be set independently, but peak can never be less than sustain.

Without an input connected, a single full envelope is generated.

## 72.2 PARAMETERS – TRIGGER PAGE

### TIE THRESHOLD LEVELS

If on, the trigger and release thresholds are the same value.

### TRIGGER THRESHOLD

The trigger threshold (see above)

### RELEASE THRESHOLD

The release threshold (see above)

### RE-TRIGGER DELAY

The amount of time after a trigger point that a new trigger may occur.

### COMPLETE ENVELOPE

If on, a complete envelope is produced for each trigger point. If off, the envelope may be terminated at any time by a release point.

### REMAINDER

See *Remainder Options* p. 305.

#### SAMPLE RATE

Sample rate when no inputs connected.

## 72.3 PARAMETERS – ATTACK PAGE

#### DELAY LENGTH

The amount of time to delay the envelope after the trigger point.

#### ATTACK LENGTH

The amount of rise time from zero to the peak level.

#### ATTACK SHAPE

The shape of the attack ramp.

#### PEAK LEVEL

The peak level.

#### PEAK LENGTH

The length of time of the peak.

## 72.4 PARAMETERS – SUSTAIN PAGE

#### DECAY LENGTH

The amount of decay time from the peak level to the sustain level.

#### DECAY SHAPE

The shape of the decay ramp.

#### SUSTAIN LEVEL

The sustain level. This level is held until a release point is reached.

#### RELEASE LENGTH

The amount of release time from the sustain level to zero.

#### RELEASE SHAPE

The shape of the release ramp.

## 72.5 STANDARD OPTIONS AND LOCAL VARIABLES


- *Remainder Options* p. 305
- *Scope and Channel Name Matching Options* p. 296
- *Units Options* p. 297
- *Local Variables* p. 306 – $C, $NC

# 73 TRIM CHOP

## 73.1 DESCRIPTION

This CHOP shortens or lengthens the input's channels. A part of the interval can be preserved or removed. If the channels are being lengthened, the extend conditions of the channel will be used to get the new values.

The handles on the Trim CHOP in the graph can adjust its length.

## 73.2 PARAMETERS

### UNIT VALUES

Determines whether the Start/End parameters are absolute numbers or numbers that are relative to the start and end of the input channels.

*Absolute* — The Start and End are the actual numbers defining the new interval.

*Relative to Start/End* — The Start and End are relative to the start and end of the input CHOP.

*Use Current Frame* — A CHOP with only one sample is output, containing the channel values at the current frame.

### START / END

The start / end of the range to trim.

### DISCARD

Which part of the channel to discard:

*Exterior* — Discard those parts of the channel outside the trim range.

*Interior* — Discard the interior of the trim range.
If two intervals remain, sequence them together.

The CHOP is shortened by increasing the Start relative value, and lengthened by decreasing it. The CHOP is lengthened by increasing the End relative value, and shortened by decreasing it.

See the *Extend CHOP* p. 340 for more details on the extend conditions.

## 73.3 STANDARD OPTIONS AND LOCAL VARIABLES

- *Units Options* p. 297
- *Start-End Options* p. 302
- There are no local variables.

# 74 VOICE CHOP

## 74.1 DESCRIPTION

The Voice CHOP detects phonemes in an audio channel given a previous series of audio phoneme samples.

## 74.2 PARAMETERS

### QUALITY

Choose the quality / speed you would like to process by; faster or better.

### SILENCE LEVEL

A higher value tolerates more ambient noise.

Silence Channel Name

Name for silence channel.

### VOCAL RANGE

Allows you to specify the frequency range of the vocal, in order to filter out other ambient sounds in the audio.

### MAX. PITCH SHIFT

Allows you to determine the extents of the pitch shifting allowed within a phoneme.

### NOISE FILTER (HZ)

Sets a noise filter on the input signal – set to get a clearer sample for speech.

### SETUP PHONEME LIBRARY

Allows you to setup a series of samples to determine your base phonemes by.

# 75 WARP CHOP

## 75.1 DESCRIPTION

This CHOP time-warps the channels of the first input (the Pre-Warp Channels) using one warping channel in the second input (the Warp Curve). The Warp Curve acts as either a rate control or an index control, as explained below.

In the Rate Control Method, feeding the Warp CHOP a Warp Curve with a constant value of 1 makes the output identical to the Pre-Warp Channels, assuming the two inputs have the same start-end interval. That is, where the rate is 1, the Pre-Warp Channels are not warped.

Where the Warp Curve is above 1, it causes a speed-up in the animation. Where the Warp Curve is below 1, it causes a slow-down in the animation. Rates less than 0 cause the animation to go in reverse.

In the *Index Control* Method, the Warp Curve acts as an index into the first input. If the Warp Curve is a straight ramp with a slope of 1 (in Units), it produces unwarped output channels. If Units is set to Seconds, a Warp Curve value of 0 gets the Pre-Warp Channels' values at time 0 seconds. A Warp Curve value of 2 gets the Pre-Warp Channels' values at time 2 seconds.

The Warp CHOP will output the same number of channels and channel names as the Pre-Warp Channels input, and the sample rate will be the same as that of the Pre-Warp input. However, the CHOP will output the same start-end time interval as the Warp Curve input.

If you take a Warp Curve and pass it directly to a Warp CHOP with the Rate Control Method, it is equivalent to passing the same curve to an Area CHOP and then passing it to the Warp CHOP with the Index Control Method.

## 75.2 PARAMETERS

### METHOD

The warping method to use: *Rate* or *Index Control*.

### STRETCH INDICES TO CHANNEL LENGTH

If on, the minimum and maximum values in the Warp Curve are mapped to the beginning and end of the channels to be warped. Otherwise, the Warp Curve is applied as-is to the Pre-Warp Channels.

The Warp CHOP does linear interpolation of the Pre-Warped Channels when they are both stretched and compressed.

You can also warp a repeating set of channels with the Oscillator CHOP.

## 75.3 STANDARD OPTIONS AND LOCAL VARIABLES

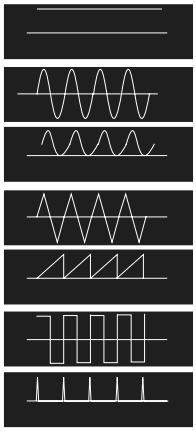
- *Scope and Channel Name Matching Options* p. 296
- *Sample Rate Options*
- *Units Options* p. 297
- There are no local variables.

# 76 WAVE CHOP

## 76.1 DESCRIPTION

The Wave CHOP creates a waveform that is repeated.

## 76.2 PARAMETERS – WAVEFORM PAGE



### TYPE

There is a choice of waveforms with different value-ranges:

| | |
|---|---|
| *Constant* (1) | A constant-valued "wave". |
| *Sine* (-1 to 1) | A Sine wave. |
| *Gaussian* (0 to 1) | A Gaussian (also known as a bell or normal) curve. |
| *Triangle* (-1 to 1) | A ramp-up and down that can be shaped with *Bias*. |
| *Ramp* (0 to 1) | A ramp in 0 to 1 range. |
| *Square* (-1 to 1) | Step-up/step-down, with Bias control to make it pulse. |
| *Pulse* (0 to 1) | Produces a 1 for one frame, 0 otherwise. |
| *Expression* | Works with the Expression parameter. |

The Sine formula is:

```
out = Amplitude * (Offset + sin ((Period*index - Phase)*360))
```

### PERIOD

The period is the number of seconds, frames or samples that the waveform repeats in. It is expressed in the CHOP's Units (default is Seconds), found on the *Common* page.

### PHASE

The phase shifts the waveform in time, and is expressed as a fraction of a period, usually between 0 and 1.

### BIAS

You can vary the shape of some of the waveform types by changing the bias within the range -1 to +1.

#### OFFSET

The waveform's value can be offset. A sine wave can remain always positive by setting Offset to 1.

#### AMPLITUDE

The wave's value can be scaled.

#### DECAY RATE

The wave's amplitude can be reduced over time with an "exponential decay". For example, if the Decay is 0.2 and the *Units* are seconds, then the amplitude will decay to 0.8 after 1 second, and 0.8 of 0.8 (or 0.64) after 2 seconds.

#### RAMP SLOPE

Then a ramp is added to the result with a slope of Ramp. The channel increases by the Ramp Slope value every Unit of time. For example, if Ramp is 1.2, the channel increases by 1.2 every second, in addition to the shape of the wave.

#### EXPRESSION

If the waveform type is *Expression*, the *Expression* parameter is used to input a math expression. Some local variables are available: `$I` (Index), `$L` (the loop variable over the period 0 to 1), `$C` (the cycle variable, the integer number of cycles the waveform has passed at the current index).

### 76.3 PARAMETERS – CHANNEL PAGE

#### CHANNEL NAME

You can creates many channels with simple patterns like "`chan[1-20]`". See the section, *Common CHOP Parameters* p. 296 for a description of this and all Options. The CHOP uses the local variable, $C – the channel number – which can be used to make each channel different. See *Scope and Channel Name Matching Options* p. 296.

#### START / END

Start and end of interval, expressed in *Units*.

## 76.4 STANDARD OPTIONS AND LOCAL VARIABLES

- *Units Options* p. 297
- *Scope and Channel Name Matching Options* p. 296
- *Extend Options* p. 303
- *Local Variables* p. 306 –$C, $NC, $I and:

| | |
|---|---|
| $L | The loop index, used to determine the location of the current index within the current cycle. It ranges between 0 and the Period-1, in samples. |
| $N | The cycle index, which is the number of complete cycles that have occurred up to the current index. |
| $P | The value of the period parameter, in samples. |
| $PH | The value of the phase parameter, in samples. |
| $B | The value of the bias parameter, in samples. |

# 3 Expressions, Stand Alones and Formats

## 1 CHOP EXPRESSION FUNCTIONS

### 1.1 EXPRESSION FUNCTIONS ACCESSIBLE FROM ANYWHERE IN HOUDINI

*chop("choppath/channelname")*
This evaluates `choppath/channelname` at the current time/frame. An example is `"/ch/ch1/wave1/chan1"`.

*chopf("choppath/channelname", frame)*
This evaluates the channel at the specified `frame`.

*chopt("choppath/channelname", time)*
This evaluates the channel at the specified `time`, expressed in seconds.

*chopi("choppath/channelname", index)*
This evaluates the channel at the specified *index* .

*chopcf("choppath", channelnum, frame)*
Like *chopf()* this evaluates the channel at the specified *frame*, but the channel is specified with two fields: the CHOP name and the channel number, as an index that starts at 0. Example: *chopcf("/ch/ch1/wave1", 0, 61)* .

*chopct("choppath", channelnum, time)*
Like *chopt()* this evaluates the channel at the specified *time*, but the channel is specified with two fields: the CHOP name and the channel number.

*chopci("choppath", channelnum, index)*
Like *chopi()* this evaluates the channel at the specified *index*, but the channel is specified with two fields: the CHOP name and the channel number.

*chopstr("choppath/channelname")*
Like *chop()*, this evaluates a CHOP output channel at the current time/frame. It doesn't return a float (a raw number) like the other functions, but a string in ASCII containing the same number. (Both output the same, so they appear similar.)

*chopn("choppath")*
This returns the number of channels in the CHOP.

*chops("choppath")*
This returns the start index of CHOP, expressed as samples. To express the start of the CHOP in seconds, divide this by *chopr()* .

*chope("choppath")*
This returns the end index of CHOP.

*chopl("choppath")*
This returns the length of the CHOP in samples.

*chopr("choppath")*
This returns sample rate of the CHOP.

### EXAMPLE

To get the value of a channel from a Wave CHOP at the previous frame:

As it is in UNIX, `/` is the root of all data, and in Houdini, */ch* is where the CHOP Networks are located. So if the first CHOP Network is *ch1*, and the Wave CHOP within it is called *wave1*, you need: */ch/ch1/wave* .

But this only identifies the CHOP, not its channels. So you need to append the channel you want to this, resulting in: */ch/ch1/wave/chan1* . So the channel function is:

```
chopf("/ch/ch1/wave1/chan1", $F-1)
```

## 1.2 FUNCTIONS LOCAL TO CHOPS

If you are working within the context of CHOPs – say by putting math expressions in the Expression CHOP and fetching channels from Expression's input – you should use the faster functions:

*beat(keyname parameter)*
Get the value attached to one of the keyboard keys. The first argument is the key name, including the 0 to 9 keys, named *KEY0* to *KEY9*, the A to Z keys named *KEYA* to *KEYZ*, and the keypad keys named *KEYPAD0* to *KEYPAD9*. The second argument is the parameter to retrieve: VAL, SPEED, BINARY, ANALOG, TICK, COUNT, PERIOD, CYCLE.

*icn(input)*
Returns the number of input channels at *input*, where 0 = first input to the CHOP.

*ics(input)*
This returns the 'start index' of the *input*.

*ice(input)*
This returns the 'end index' of the *input*.

*icl(input)*
This returns the length of the *input*, expressed in samples.

*icr(input)*
This returns the sample rate of the *input*.

*icmax(input, channelindex)*
This returns the maximum value in the channel *channelindex* of *input*.

*icmin(input, channelindex)*
This returns the minimum value in the channel *channelindex* of *input*.

*ic(input, channelindex, sampleindex)*
This returns the value of the channel number *channelindex*, of input number *input*, at sample number *sampleindex*. They all start at 0.

*oc(channelindex, sampleindex)*
This gets values from the output channel as the CHOP's output is being calculated. e.g. While the Expression CHOP is computing its output at index `$I`, it can access the output values at the previous index, `$I-1`. This is useful when stepping forward frame-by-frame. The *oc()* function is only valid for *sampleindex* < *$I*.

***Note:*** *ics, ice, icl, chops, chope, chopl* return values in terms of 'sample index' and represent the entire CHOP.

## 1.3 USEFUL REGULAR EXPRESSION FUNCTIONS

### OPDIGITS()

The expression function, *opdigits()* gets the numbers that are in an OP name. It gets *23* from and OP named *leg23right* .

This allows animators to make one CHOP Network for each of 100 similar characters. For example, in a CHOP, you can get the number of the CHOP Network:

*opname("..")* = *ch1* (a string)
*opdigits("..")* = *1* (a set of digits)

`"."` is name of the OP that the expression is in.
`".."` is the parent OP. If it occurs in a CHOP, `".."` means the CHOP network it is in.

`opdigits(string name)`
This function returns a value of the concatenation of all the digits in a node's name. It is used when building several similar networks. For example:

```
opdigits("/obj/geo7") = 7

opdigits("..") = 7        (at the SOP level of geo1)
```

To make a channel name that is for export to an object, and you want to name the channels to match the CHOP network to the object name, create a name that look like this:

```
geo`opdigits("..")`:tx
```

If this is in a CHOPnet called *ch3*, the channel name will be *geo3:tx* .

### MODBLEND()

The *modblend()* function blends between two numbers using a weighting factor.

# 2 STAND ALONE CHOP PROGRAMS

## 2.1 INTRODUCTION

These are stand-alone commands that can be run in the UNIX or NT shell.
For complete information, see *Stand Alone > Channel Tools (chTools)* p. 425.

## 2.2 CHTOOLS

| | |
|---|---|
| *chchan* | Copy channel collection to/from row/column channel format. |
| *chcp* | Copy channel collection file to another format. |
| *chinfo* | Display information on a channel collection. |
| *claudio* | Copy CHOP data (clip) to/from an audio format. |
| *clchan* | Copy CHOP data (clip) to/from action channel format. |
| *clchn* | Copy CHOP data (clip) to/from channel collection . |
| *clcp* | Copy CHOP data to another format. Converting to chan loses information. |
| *clinfo* | Display information on CHOP format. |

## 2.3 COMMAND LINE OPTIONS

For more information, run each command with the **-** argument. For example:

```
claudio -
```

# 3 CHOP FILE FORMATS

## 3.1 CHOP FORMATS

| Extensions | Type | Read | Write | Notes |
|---|---|---|---|---|
| .chan | External | • | • | Houdini ASCII Channel Format containing raw values in rows & columns. |
| .bchan | External | • | • | Houdini Binary Channel Format, equivalent to .chan |
| .clip | Internal | • | • | Houdini ASCII Native CHOP Format. It contains raw values like .chan plus more information. |
| .bclip | Internal | • | • | Houdini Binary Native CHOP Format, equivalent to .clip |
| .chn | External | • | • | Houdini ASCII Format for a group of channels expressed as keyframed segments (Bézier, ease, etc.) |
| .bchn | External | • | • | Houdini Binary Format, equivalent to .chn |
| .aiff | Internal | • | • | Audio Format |
| .aifc | Internal | • | • | Audio Format |
| .au | Internal | • | • | Audio Format |
| .sf | Internal | • | • | Audio Format |
| .snd | Internal | • | • | Audio Format |
| .wav | Internal | • | • | Audio Format |

External formats are implemented as standalone converter programs that are piped into Houdini. Internal formats are implemented as converters that are built-into the Houdini program itself. Both format types appear the same to Houdini users. Internal formats are slightly faster. New external formats can be interfaced to Houdini without requiring the Houdini Developers' Kit, only the public-domain source code as described below.

## 3.2 FOR FURTHER INFO

See the *Formats > Channel File Formats* p. 257 section.