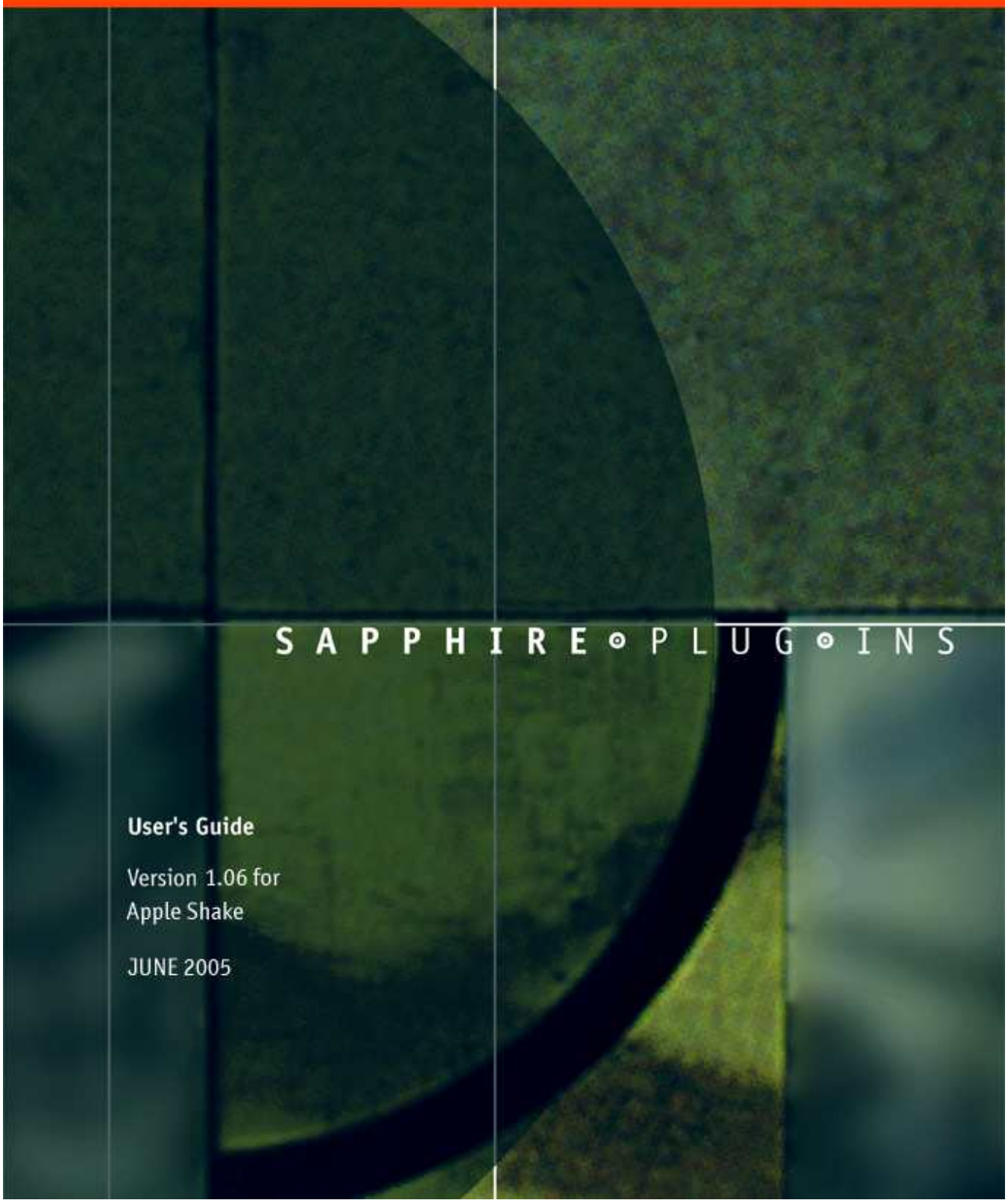
GENARTS



SAPPHIRE PLUG-INS

User's Guide

Version 1.06 for
Apple Shake

JUNE 2005

Table of Contents

Table of Contents

Table of Contents

Table of Contents

Preface

Contact Information

GenArts, Inc.
8 Clinton Street
Cambridge, MA 02139−2304
USA

Tel: 1−617−492−2888
Fax: 1−617−492−2852
Email: info@genarts.com
web.genarts.com

License Agreement

If you install and/or use this software, you agree to be bound by the terms and conditions of this agreement.

You are hereby granted a non−exclusive, non−transferable license to use this software on a temporary trial basis. If purchased, you may continue using it on specific computers under these same conditions.

This software is owned by GenArts, Inc., and is protected under copyright law. However, you may copy it onto and amongst workstations running Shake at your facility.

You may not reverse engineer, decompile, or disassemble this software.

You should test this software thoroughly before depending on it, since like any software it may contain bugs. Although efforts will be made to document and/or repair problems as they are found, this software and documentation are provided "as is" without warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the results and performance of the software is with you. In the event of trouble, your exclusive remedy shall be repair or replacement of this software. GenArts, Inc. will not be liable for any consequential, incidental, indirect or special damages arising from the use of this software under any theory of liability, whether contract, strict liability or tort including negligence or otherwise. In any event, GenArts' total liability shall not exceed the amount paid by you for this software.

Copyrights

Introduction

Sapphire Plug−ins version 1 is a package of image processing and synthesis effects for use with Shake 3.5 or greater.

It includes 58 plug−ins, each with many options and parameters which can be adjusted and animated for an unlimited range of results.

For updated information please check web.genarts.com. Also see our Shake support page for help with technical issues.

What's new in this version

The following are new since version 1.0:

1. A problem with link−pasting Sapphire Plug−in nodes (with Shift−Ctrl−V) has been fixed.
2. A problem with creating macros with Sapphire Plug−in nodes has been fixed.
3. In 1.02, a problem with with multiple warp effects in Shake 3.01, and an issue with the installer clearing existing icons has been corrected.
4. In 1.03, Shake 3.5 is supported, performance has been improved, and miscellaneous issues have been fixed.
5. In 1.04, performance of Rack Defocus has been improved, and further miscellaneous issues have been fixed.
6. In 1.05, Mac OS X Tiger (10.4) and Shake 4.0 are supported. Some issues with S_MatteOps have also been fixed.
7. In 1.06, multiple Shakes running on a single machine only take a single Sapphire license. Also, rebooting Shake client machines will no longer cause the Sapphire FLEXlm server to exit.

Loading a Plug−in

To load a plug−in, click on the GenArts tool tab in the tool menu, and a menu of all Sapphire Plug−ins should appear. You can use a plug−in just like any other effect.

The Sapphire wipe and dissolve transition effects also appear in the mix menu of the standard Shake transition node.

Many Sapphire Plug−ins include several different effect variations. For example the S_Wipes plug−in contains 17 different kinds of Wipe transitions. For these plug−in, a popup menu or radio buttons appear at the top of the tweaker, which allows selecting between the different effect options. The name of this parameter is usually Effect but is sometimes more specific such as Mode, Operation, Shape, or Pattern. The Load Defaults button does not change this parameter.

Load Defaults

There are two ways to reset a plug−in's parameters to their default values:

You can right click on the parameter page and select Reset All. This will re−initialize the plug−in from scratch, and is equal to deleting the plug−in and loading a new copy of it.

All plug−ins also include a Load Defaults button at the bottom of their parameter page. When pressed, this resets the parameters to their default values, but it does not change the currently selected Effect option, and does not close any parameter groups that may be open.

Online documentation

All Sapphire Plug−ins include a "Help" button in their parameter page. When pushed, this should show the on−line doc for that plug−in in your HTML browser.

Online documentation is normally installed along with your software and can also be accessed directly. On Mac, go to the Applications folder, double click on GenArtsSapphireShake, and select View Online Documentation. On Linux or SGI Irix, point your HTML browser at /usr/genarts/sapphire−nreal/docs/intro.html.

About Mask inputs

Shake provides optional masking for all plug−ins. the processed result appears where the Mask is white, and the unprocessed first input (usually the background) shows through where the Mask is black.

Some Sapphire Plug−ins also accept their own S_Mask input and perform their own method of Mask processing.

Glint, Glow, and Glare, for example, take both a Source and an optional S_Mask input, and multiply the source input by the mask *before* generating the glints (or glows, glares), so where the mask is black no glints are generated, and where it is white they are generated as usual. This method prevents the glints or glows themselves from being partially cropped by the mask. In addition these effects use the RGB colors of the S_Mask input to selectively colorize the resulting glows, glints, or glares. The red areas of the mask will produce red glows, glints, or glares, and so on.

In Blur, the difference between Shake masking and the masking performed by the plug−in is more subtle. The areas which are masked out are never blurred, so they do not blur into the masked−in regions. If the mask were applied afterward, the pixels behind the matte would be blurred over the edge of the matte and into the final image. As an example, say you have a clip with white text over a black background. If you put that clip into both the Source and S_Mask inputs of Blur, the black background will *not* be blurred into the text, since the black pixels are all matted out.

About Alpha channel processing

All Sapphire Plug−ins can handle RGBA inputs, and the Alpha of RGBA inputs is handled in one of three ways, depending on the effect:

1. Alpha is processed as just another input channel like R, G, and B. Effects in this category include: AutoPaint, Mosaic, Blur, BlurMotion, RackDefocus, all Wipes, all Dissolves, Distort, DistortBlur, DistortChroma, all Kaleidoscopes, all Warps, Shake, and MathOps.

2. Alpha is copied from the first input to the output. In this case the effect doesn't use the Alpha channel, but it is passed through unchanged from the first input to the output. Effects in this category include: BandPass, BlurChroma, ClampChroma, DuoTone, EdgeDetect, Embosses, Etching, HalfTones, Hotspots, DistortRGB,

Monochrome, Pseudo_Color, Psykos, Sharpen, Sketch, Sparkles, Streaks, Threshold, and Zebrafy. This is the same as what DS does if Process Alpha is off.

3. Some other effects pass the Alpha through by default, but include an Affect Alpha parameter for optionally adding to the alpha channel. These are LensFlare, all Glows, all Glints, Glare, and all Zaps.

If our method of Alpha processing is not appropriate for your application, you can usually manipulate Alpha to be what you need using the Channel Switcher.

In some cases, it is not obvious how Alpha should be processed. When possible we've chosen the method that seems best for use with 'premultiplied' compositing. This is often also called 'additive' compositing and is done when the R, G and B values have already been multiplied by Alpha:

```
Result = Background * (1 − Alpha) + Foreground
```

For example, by default LensFlare, Glows, Glints, and Glares pass the background's Alpha through to the output without modification. This is because these effects simulate light emission. They are purely additive and have no opacity themselves. You should get the correct effect if you composite the output of LensFlare (applied to an RGBA clip) over another clip using a premultiplied composite. The original clip will composite over the background using its Alpha, and the flare will simply add to the result.

Effects with an Affect Alpha parameter can optionally increase the output Alpha where the flares (or glows, glints, or glares) occur, but in this case the flare elements will become opaque and will darken the background image behind them when composited later.

Parameters and Viewport dimensions

Many Sapphire Plug−in parameters affect the size of something in approximate pixels. The default value of this type of parameter is usually an expression that is a fraction of the current width or height of the viewport. For example the LensFlare scaleWidths parameter defaults to ".2*width". This causes the defaults to give resolution independent results − if you alter the viewport dimensions the LensFlare size will stretch to fill the same fraction of the viewport. However, once this parameter value is changed to a numerical value, the size will instead remain at a fixed number of pixels.

Other Sapphire parameters specify xy screen coordinates or shift amounts, such as the LensFlare pivot and Clouds shiftStart parameters. These also default to a give a resolution independent result such as at the screen center, and also remain at a fixed pixel location once they are set to numerical values. These parameters are usually shown by a crosshair on the screen which can be dragged to change their values.

Another type of parameter indicates the "frequency" of something. For example, HalfTone has a dotsFrequency parameter which determines the number of dots across the screen. This type of parameter defaults to a numerical value and is always resolution independent − if you alter the viewport dimensions the pattern will stretch accordingly. If you do want the pattern to remain the same across a fixed number of pixels instead of across the viewport, you could set the value to an expression such as "10 * 720/width" which will scale the frequency by the inverse of the viewport width, in this case giving 10 dots per 720 pixels.

About pixel aspect ratios

For some image formats, the digital form of the image is scaled non−uniformly to produce the final viewed picture. For example NTSC resolution is normally 720x486 with an aspect ratio of 1.481. However, the final NTSC picture has an aspect ratio of 1.333. Thus the original digital image is scaled in the horizontal direction by a factor of 0.9 and shapes rendered as circles can end up squashed slightly into ovals. The original pixels are effectively rectangular shaped instead of squares, and have an aspect ratio of 1.481/1.333 = 1.111.

Shake has its own *defaultAspect* value which can be adjusted in **Globals** under **format**. This value is used to default the

parameters of Sapphire Plug−ins to give the appropriately scaled results when you first load a plug−in. Many plug−ins have their own pixelAspect parameter which defaults to *defaultAspect*. Other effects may have a relative y width parameter which defaults to the inverse of *defaultAspect*.

In addition, all Sapphire Plug−ins compensate for Shake's current *proxyRatio* value, each time they render.

If necessary, you can also scale the pixel aspect ratio for *all* Sapphire Plug−ins by changing the value of scale_pixel_aspect_ratio in the "s_config.text" file.

The pixel aspect ratio makes no difference for basic pixel processing effects such as color processing or compositing.

Script Forms

The script forms of Sapphire Plug−ins look like this:

```
S_PluginName(Input1, Input2, version, Effect, param1, param2...);
```

The image inputs are listed first, followed by the Sapphire `version` number for future compatibility. Next is the `Effect` option for those plug−ins that have one, which is a string for popup menus, or a number for radio buttons. Finally the parameters are listed. The documentation pages for each effect show these parameters in order. Two dimensional XY parameters become two floats like `paramX, paramY` in script form, and color parameters become three floats, such as `paramRed, paramGreen, paramBlue.` The initial part of the script form of each effect is shown at the bottom of its documentation page.

Customizing Plug−ins

A facility is included with Sapphire Plug−ins that allows users with some programming experience to define and customize new plug−ins. A number of parameters are also available that can be adjusted to customize the behavior of all Sapphire plug−ins. You can disable multi−processing, or specify lookup tables for more accurate processing of log format images. For additional information on these, or to modify a parameter, see the s_config.text file.

To open the config file on Mac, go to the Applications folder, control click on GenArtsSapphireShake, select Show Package Contents, double click on Contents, then double click on s_config.text.

On Linux or SGI Irix the config file is located at `/usr/genarts/sapphire−nreal/s_config.text`.

Known problems

1. If a plug−in has no inputs connected, and interactiveScale is less than 1, the on−screen cross hairs can not be interactively dragged if their value is an expression that includes "width" or "height". A workaround is to adjust the the parameter with the slider first. Hopefully this will be fixed in a future version of Shake.

2. If a plug−in has no inputs connected, and interactiveScale is less than 1, the viewer resolution can temporarily pop larger and smaller when you press and release a slider. Hopefully this will be fixed in a future version of Shake.

3. If the input image of a plug−in node is smaller than its output, and you view the output while tweaking the input, only a subset of the viewed image will be updated unless the "incremental update" is off. Hopefully this will be fixed in a future version of Shake.

4. If you include a Sapphire Plug−in in a Shake macro and you want to expose a plug−in's popup button as a macro parameter, you need to quote and prepend a colon to the name in the macro .h file (`name` should become `":`

name"). Also, for any exposed popup−menu or radio−button parameters, you need to add their UIs manually to the macro UI .h file. You can visit our support page for more details on how to do this. Hopefully these workarounds will not be necessary in a future version of Shake.

5. Load Defaults can not be undone properly. Hopefully this can be fixed in a future version of Shake.

6. Z channel data is not accessed or processed by Sapphire Plug−ins. It is only copied from the input to the output unmodified.

7. The Feedback effects do not work when rendering in field mode because plug−ins are not given fields in chronological order.

8. Rendering in field mode is sometimes not as efficient as it should be due to the way Shake renders fields.

9. The color parameters of plug−ins contain only RGB sliders instead of all the "RGBHSVTML" options that many Shake colors have. This is due to a Shake limitation which will hopefully be fixed in a future version.

10. Most of our on−screen user interface "widgets" can not be supported due to limitations of Shake's plug−in api. We hope to support these in future versions of Shake.

Acknowledgements

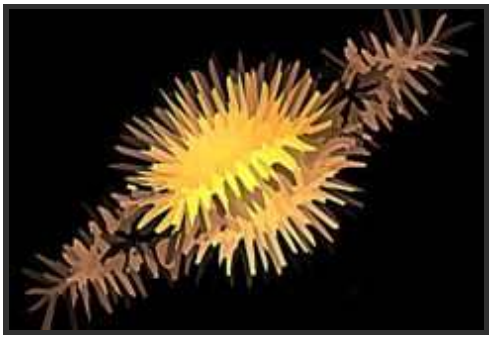
We are grateful to our many customers who have made suggestions and taken time to beta test this software. The software for JpegDamage is based in part on the work of the Independent JPEG Group.

Effects

The remainder of this User's Guide contains information about each effect in the Sapphire Plug−ins package. The effects are listed in alphabetical order by plug−in name.

Each effect's documentation describes the functionality of the effect, its inputs and parameters, and contains an example picture. This same documentation for each effect is also available directly from the Plug−in interface by clicking on the **Help** button while using any effect.

AutoPaint: HairyPaint



In the S_AutoPaint Plugin.

Generates a 'paint−brushed' version of the source clip. Similar to VanGogh but the strokes are aligned perpendicular to the edges within the image.

Inputs:

Source: The clip to be processed.

Parameters:

frequency: *Default:* 50, *Range:* 1 or greater.
The density of brush strokes in the frame. Increase for smaller strokes.

strokeLength: *Default:* 2, *Range:* any.
Determines the length of the brush strokes along the directions of edges in the source clip. If this is negative you can switch from VanGogh to HairyPaint styles and vice versa.

strokeAlign: *Default:* 0.2, *Range:* 0 or greater.
Increase to smooth out the directions of the strokes so nearby strokes are more parallel.

smoothColors: *Default:* 0, *Range:* 0 or greater.
Blurs the source by this amount before generating the brush strokes. Increase to cause the colors of nearby strokes to be more consistent.

seed: *Default:* 0, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

jitterFrames: *Integer*, *Default:* 0, *Range:* 0 or greater.
If this is 0, the locations of the strokes will remain the same for every frame processed. If it is 1, the locations of the stokes are re−randomized for each frame. If it is 2, they are re−randomized every second frame, and so on.

sharpen: *Default:* 1, *Range:* any.
The amount of post−process sharpening applied.

sharpenWidth: *Default:* 0.1, *Range:* 0 or greater.
The width at which to apply the post−process sharpening filter, relative to the stroke sizes. Higher values affect wider areas from the edges, lower values only affect areas near sharp edges.

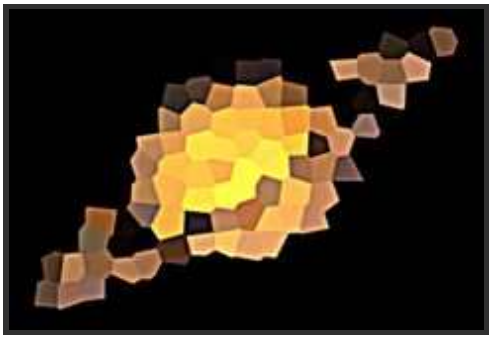
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels. This parameter should not be animated for this effect because the results will pop around the value of 1.5.

Script Form: `S_AutoPaint(Source, 1.06, "Hairy Paint", frequency, etc...);`

See Also:

VanGogh
Pointalize
Sketch
Etching
HalfTone
HalfToneColor

AutoPaint: Pointalize



In the S_AutoPaint Plugin.

Generates a 'paint−brushed' version of the source clip. Similar to VanGogh but the strokes are cellular pointy shapes with no direction.

Inputs:

Source: The clip to be processed.

Parameters:

frequency: *Default:* 50, *Range:* 1 or greater.
The density of brush strokes in the frame. Increase for smaller strokes.

smoothColors: *Default:* 0, *Range:* 0 or greater.
Blurs the source by this amount before generating the brush strokes. Increase to cause the colors of nearby strokes to be more consistent.

seed: *Default:* 0, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

jitterFrames: *Integer*, *Default:* 0, *Range:* 0 or greater.
If this is 0, the locations of the strokes will remain the same for every frame processed. If it is 1, the locations of the stokes are re−randomized for each frame. If it is 2, they are re−randomized every second frame, and so on.

sharpen: *Default:* 1, *Range:* any.
The amount of post−process sharpening applied.

sharpenWidth: *Default:* 0.1, *Range:* 0 or greater.
The width at which to apply the post−process sharpening filter, relative to the stroke sizes. Higher values affect wider areas from the edges, lower values only affect areas near sharp edges.

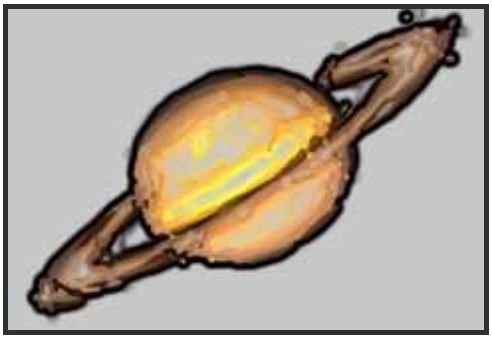
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels. This parameter should not be animated for this effect because the results will pop around the value of 1.5.

Script Form: `S_AutoPaint(Source, 1.06, "Pointalize", frequency, etc...);`

See Also:

VanGogh
HairyPaint
Sketch
SketchBumpy
Etching
HalfTone
HalfToneColor
Mosaic
FlysEyeHex
Sharpen
Sapphire Plug−ins Introduction

AutoPaint: Sketch



In the S_AutoPaint Plugin.

Generates a version of the input with a hand drawn sketched look. The results of this effect can depend on the image resolution, so it is recommended to test your final resolution before processing a clip.

Inputs:

Source: The clip to be processed.

Parameters:

frequency: *Default:* 50, *Range:* 1 or greater.
The density of brush strokes in the frame. Increase for smaller strokes.

strokeLength: *Default:* 2, *Range:* any.
Determines the length of the strokes along the directions of edges in the source clip. If this is negative you can switch between Sketch and BumpySketch styles and vice versa.

strokeAlign: *Default:* 0.2, *Range:* 0 or greater.
Increase to smooth out the directions of the strokes so nearby strokes are more parallel.

smoothColors: *Default:* 0, *Range:* 0 or greater.
Blurs the source by this amount before generating the brush strokes. Increase to cause the colors of nearby strokes to be more consistent.

seed: *Default:* 0, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

jitterFrames: *Integer*, *Default:* 0, *Range:* 0 or greater.
If this is 0, the locations of the strokes will remain the same for every frame processed. If it is 1, the locations of the stokes are re−randomized for each frame. If it is 2, they are re−randomized every second frame, and so on.

backgroundColor: *Default rgb:* [0.8 0.8 0.8].
The color of the background over which the sketched lines are applied.

lineThickness: *Default:* 0.04, *Range:* 0 or greater.
The thickness of the sketched lines.

lineStrength: *Default:* 0.3, *Range:* 0 or greater.
The strength of the sketched lines. Increase for brighter lines, decrease for softer lines.

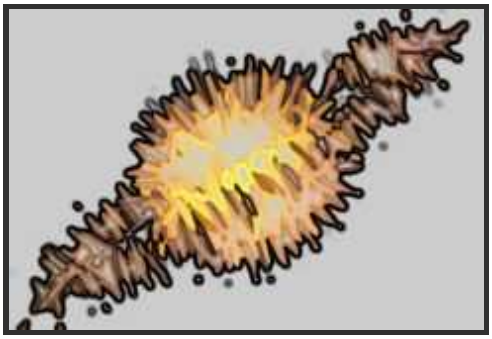
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels. This parameter should not be animated for this effect because the results will pop around the value of 1.5.

Script Form: `S_AutoPaint(Source, 1.06, "Sketch", frequency, etc...);`

See Also:

VanGogh
HairyPaint
Pointalize
SketchBumpy
HalfTone
Mosaic
FlysEyeHex
EdgeDetect
JpegDamage
Sapphire Plug−ins Introduction

AutoPaint: SketchBumpy



In the S_AutoPaint Plugin.

Generates a version of the input with a hand drawn sketched look. Similar to Sketch but the strokes are aligned perpendicular to the edges within the image.

Inputs:

Source: The clip to be processed.

Parameters:

frequency: *Default:* 50, *Range:* 1 or greater.
The density of brush strokes in the frame. Increase for smaller strokes.

strokeLength: *Default:* 2, *Range:* any.
Determines the length of the strokes along the directions of edges in the source clip. If this is negative you can switch between Sketch and BumpySketch styles and vice versa.

strokeAlign: *Default:* 0.2, *Range:* 0 or greater.
Increase to smooth out the directions of the strokes so nearby strokes are more parallel.

smoothColors: *Default:* 0, *Range:* 0 or greater.
Blurs the source by this amount before generating the brush strokes. Increase to cause the colors of nearby strokes to be more consistent.

seed: *Default:* 0, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

jitterFrames: *Integer*, *Default:* 0, *Range:* 0 or greater.
If this is 0, the locations of the strokes will remain the same for every frame processed. If it is 1, the locations of the stokes are re−randomized for each frame. If it is 2, they are re−randomized every second frame, and so on.

backgroundColor: *Default rgb:* [0.8 0.8 0.8].
The color of the background over which the sketched lines are applied.

lineThickness: *Default:* 0.04, *Range:* 0 or greater.
The thickness of the sketched lines.

lineStrength: *Default:* 0.3, *Range:* 0 or greater.
The strength of the sketched lines. Increase for brighter lines, decrease for softer lines.

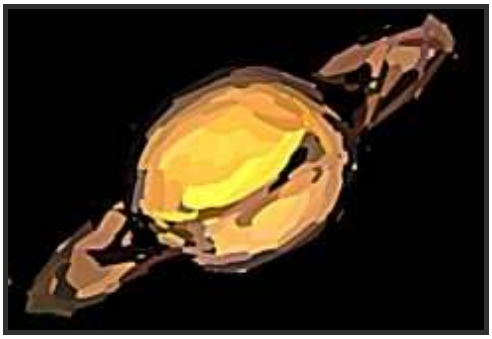
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels. This parameter should not be animated for this effect because the results will pop around the value of 1.5.

Script Form: `S_AutoPaint(Source, 1.06, "Sketch Bumpy", frequency, etc...);`

See Also:

VanGogh
HairyPaint
Pointalize
Sketch
HalfTone
Mosaic
FlysEyeHex
EdgeDetect
JpegDamage
Sapphire Plug−ins Introduction

AutoPaint: VanGogh



In the S_AutoPaint Plugin.

Generates a 'paint−brushed' version of the source clip. The paint stroke directions align with the edges found within the image.

Inputs:

Source: The clip to be processed.

Parameters:

frequency: *Default:* 50, *Range:* 1 or greater.
The density of brush strokes in the frame. Increase for smaller strokes.

strokeLength: *Default:* 2, *Range:* any.
Determines the length of the brush strokes along the directions of edges in the source clip. If this is negative you can switch from VanGogh to HairyPaint styles and vice versa.

strokeAlign: *Default:* 0.2, *Range:* 0 or greater.
Increase to smooth out the directions of the strokes so nearby strokes are more parallel.

smoothColors: *Default:* 0, *Range:* 0 or greater.
Blurs the source by this amount before generating the brush strokes. Increase to cause the colors of nearby strokes to be more consistent.

seed: *Default:* 0, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

jitterFrames: *Integer*, *Default:* 0, *Range:* 0 or greater.
If this is 0, the locations of the strokes will remain the same for every frame processed. If it is 1, the locations of the stokes are re−randomized for each frame. If it is 2, they are re−randomized every second frame, and so on.

sharpen: *Default:* 1, *Range:* any.
The amount of post−process sharpening applied.

sharpenWidth: *Default:* 0.1, *Range:* 0 or greater.
The width at which to apply the post−process sharpening filter, relative to the stroke sizes. Higher values affect wider areas from the edges, lower values only affect areas near sharp edges.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels. This parameter should not be animated for this effect because the results will pop around the value of 1.5.

Script Form: `S_AutoPaint(Source, 1.06, "Van Gogh", frequency, etc...);`

See Also:

HairyPaint
Pointalize
Sketch
SketchBumpy
Etching
HalfTone
HalfToneColor
Mosaic

BandPass

Generates an X−ray−like effect using a band−pass filter. Two blurs are performed with different widths, and the result is the difference scaled and offset by a gray value. Frequencies above and below the cutoffs are attenuated, leaving only the middle band of frequencies.

Inputs:

Source: The clip to be processed.

Parameters:

blurAmount1: *Default:* .0125*width, *Range:* 0 or greater.
The width for the first blur. Sets the low frequency cutoff.

blurAmount2: *Default:* .025*width, *Range:* 0 or greater.
The width for the second blur. Sets the high frequency cutoff.

blurRel: *X & Y, Default:* [1 1/GetDefaultAspect()], *Range:* 0 or greater.
The relative horizontal and vertical blur widths. Set Blur Rel X to 0 for a vertical−only blur, or set Blur Rel Y to 0 for a horizontal−only blur.

brightness: *Default:* 3, *Range:* any.
Scales the brightness of the result.

saturation: *Default:* 1, *Range:* any.
Scales the color saturation. Increase for more intense colors. Set to 0 for monochrome.

offsetDarks: *Default:* 0.5, *Range:* any.
Adds this gray value to the darker regions of the result. This can be negative to increase contrast.

Script Form: `S_BandPass(Source, 1.06, blurAmount1, etc...);`

See Also:

EdgeDetect
EdgesInDirection
EdgeColorize
Sharpen
Emboss
Blur
Sapphire Plug−ins Introduction

Blur

In the S_Blurs Plugin.

Blurs the source clip by an arbitrary amount using a pseudo−gaussian, triangle, or box filter. This effect should render quickly even with very large Width values. Use the Blur Rel X and Y parameters for a more horizontal or vertical blur direction.

Inputs:

Source: The clip to be processed.

S_Mask: *Optional.* If connected, the blur is only performed on regions of the source clip specified by the bright areas of this input. Pixels outside this mask are not blurred, and do not contribute to the resulting blurred pixels within it. Only the alpha channel of this input is used (or green/luma if there is no alpha).

Parameters:

blurAmount: *Default:* .05*width, *Range:* 0 or greater.
Scales the width of the blur.

blurRel: *X & Y, Default:* [1 1/GetDefaultAspect()], *Range:* 0 or greater.
The relative horizontal and vertical blur widths. Set Blur Rel X to 0 for a vertical−only blur, or set Blur Rel Y to 0 for a horizontal−only blur.

scaleResult: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

offsetDarks: *Default:* 0, *Range:* any.
Adds this gray value to the darker regions of the result. This can be negative to increase contrast.

filter: *Radio buttons, Default: Gauss.*
The type of convolution filter to blur with.

Box: uses a rectangular shaped filter.
Triangle: smoother, uses a pyramid shaped filter.
Gauss: smoothest, uses a gaussian shaped filter.

subpixel: *Toggle−button, Default:* off.
Enables blurring by subpixel amounts. Use this for smoother animation of the Blur Amount or Blur Rel parameters.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Blurs(Source, S_Mask, 1.06, "Blur", blurAmount, etc...);`

See Also:

BlurChannels
BlurChroma
RackDefocus
DefocusPrism
EdgeBlur

BlurChannels

In the S_Blurs Plugin.

Blurs each channel of the source clip by an arbitrary amount using a pseudo−gaussian, triangle, or box filter. This effect should render quickly even with very large Width values. Use the Blur Rel X and Y parameters for a more horizontal or vertical blur direction.

Inputs:

Source: The clip to be processed.

S_Mask: *Optional.* If connected, the blur is only performed on regions of the source clip specified by the bright areas of this input. Pixels outside this mask are not blurred, and do not contribute to the resulting blurred pixels within it. Only the alpha channel of this input is used (or green/luma if there is no alpha).

Parameters:

blurAmount: *Default:* .05*width, *Range:* 0 or greater.
Scales the width of the blur for all channels.

blurRed: *Default:* 0, *Range:* 0 or greater.
The blur width of the red channel, relative to Blur Amount.

blurGreen: *Default:* 0.5, *Range:* 0 or greater.
The blur width of the green channel, relative to Blur Amount.

blurBlue: *Default:* 1, *Range:* 0 or greater.
The blur width of the blue channel, relative to Blur Amount.

blurAlpha: *Default:* 0, *Range:* 0 or greater.
The blur width of the alpha channel if present, relative to Blur Amount.

blurRel: *X & Y, Default:* [1 1/GetDefaultAspect()], *Range:* 0 or greater.
The relative horizontal and vertical blur widths. Set Blur Rel X to 0 for a vertical−only blur, or set Blur Rel Y to 0 for a horizontal−only blur.

scaleResult: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

scaleRed: *Default:* 1, *Range:* 0 or greater.
Scales the blurred red channel.

scaleGreen: *Default:* 1, *Range:* 0 or greater.
Scales the blurred green channel.

scaleBlue: *Default:* 1, *Range:* 0 or greater.
Scales the blurred blue channel.

scaleAlpha: *Default:* 1, *Range:* 0 or greater.
Scales the blurred alpha channel, if present.

offsetDarks: *Default:* 0, *Range:* any.
Adds this gray value to the darker regions of the result. This can be negative to increase contrast.

offsetRed: *Default:* 0, *Range:* any.
Adds this value to the red channel of the result.

offsetGreen: *Default:* 0, *Range:* any.
Adds this value to the green channel of the result.

offsetBlue: *Default:* 0, *Range:* any.
Adds this value to the blue channel of the result.

offsetAlpha: *Default:* 0, *Range:* any.
Adds this value to the alpha channel of the result, if present.

filter: *Radio buttons, Default: Gauss.*
The type of convolution filter to blur with.

> ***Box:*** uses a rectangular shaped filter.
> ***Triangle:*** smoother, uses a pyramid shaped filter.
> ***Gauss:*** smoothest, uses a gaussian shaped filter.

subpixel: *Toggle−button, Default:* off.
Enables blurring by subpixel amounts. Use this for smoother animation of any of the blur amount parameters.

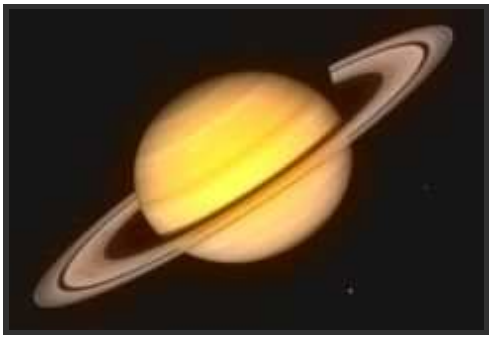
invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Blurs(Source, S_Mask, 1.06, "Blur Channels", blurAmount, etc...);`

See Also:

RackDefocus
Glow
Sapphire Plug−ins Introduction

BlurChroma



In the S_Blurs Plugin.

Separates the source into luminance and chrominance components, blurs the chrominance and/or the luminance independently, and recombines them. You can also scale the luma and chroma independently to enhance or remove either.

Inputs:

Source: The clip to be processed.

S_Mask: *Optional.* If connected, the blur is only performed on regions of the source clip specified by the bright areas of this input. Pixels outside this mask are not blurred, and do not contribute to the resulting blurred pixels within it. Only the alpha channel of this input is used (or green/luma if there is no alpha).

Parameters:

blurChroma: *Default:* .05*width, *Range:* 0 or greater.
The amount to blur the chrominance.

blurLuminance: *Default:* 0, *Range:* 0 or greater.
The amount to blur the luminance.

blurRel: *X & Y, Default:* [1 1/GetDefaultAspect()], *Range:* 0 or greater.
The relative horizontal and vertical blur widths. Set Blur Rel X to 0 for a vertical−only blur, or set Blur Rel Y to 0 for a horizontal−only blur.

scaleChroma: *Default:* 1, *Range:* 0 or greater.
Scales the chrominance by this amount. Increase for more intense colors, decrease for muted colors.

scaleLuminance: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

offsetResult: *Default:* 0, *Range:* any.
Adds this gray value to the result (or subtracts if negative). 0 has no effect, .5 is middle gray, and 1 is white.

filter: *Radio buttons, Default: Gauss.*
The type of convolution filter to blur with.

> ***Box:*** uses a rectangular shaped filter.
> ***Triangle:*** smoother, uses a pyramid shaped filter.
> ***Gauss:*** smoothest, uses a gaussian shaped filter.

subpixel: *Toggle−button, Default:* off.
Enables blurring by subpixel amounts. Use this for smoother animation of the Blur Chroma or Blur Luminance parameters.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Blurs(Source, S_Mask, 1.06, "Blur Chroma", blurChroma, etc...);`

See Also:

RackDefocus
DefocusPrism
WarpChroma
Sapphire Plug−ins Introduction

BlurMoCurves

Performs a motion blur and optionally transforms the source clip using the animated curves of the zDist, rotate and shift parameters. If these parameters are constant, no motion blur will occur.

Inputs:

Source: The clip to be processed.

Parameters:

effect: *Popup menu, Default: Transform and Blur*.
Allows disabling of the transformation.

> ***Transform and Blur:*** transforms the Source as well as blurring.
> ***Blur Only:*** this can be useful if the motions have already occurred. The curves are used only to apply the corresponding motion blur in place, and no transformation is performed.

shutterDuration: *Default:* 1, *Range:* 0 or greater.
The amount of time, in frames, to apply the motion blur over. Larger values cause more blurring, smaller values cause less. The curves are sampled at plus and minus half of this value.

shutterShift: *Default:* 0, *Range:* any.
The time−shift in frames of the motion blur. If the Shutter Speed is 1.0 and Shutter Shift is 0, the blur is calculated between the current frame −.5 and +.5. If the Shutter Shift is instead .5 then the motion blur would be calculated between the current frame and the next frame.

exposureBias: *Default:* 0.5, *Range:* 0 to 1.
Determines the variable amount of exposure along the path between the From and To transformations. A value of 0 causes more exposure at the From end, 0.5 causes equal exposure along the path, and 1.0 causes more exposure at the To end. If you have bright spots on a dark background, a 0 value would cause the processed spots to be brighter at the From end and dark at the To end, and a 1.0 value would cause the opposite.

scaleResult: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

zDist: *Default:* 1, *Range:* 0.001 or greater.
The 'distance' of the image from the camera, about the Center position. The rate of change of this parameter is also used for the motion blur. Values greater than 1.0 move it farther away and make it smaller. Values less than 1.0 move the image closer and enlarge it.

rotate: *Default:* 0, *Range:* any.
Rotates the image by this amount in degrees, about the Center. The rate of change of this parameter is also used for the motion blur. Note that for high rotation speeds, the motion blur will become less accurate.

shift_: *X & Y, Default:* [0 0], *Range:* any.
Translates the source image by this amount. The rate of change of this parameter is also used for the motion blur. It is in screen coordinates for easy use with tracker data.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The center of rotation and zooming, in screen coordinates relative to the center of the frame. The shift values should be zero for this location to make sense.

wrap: *X & Y, Radio buttons, Default:* [*No No*].
Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

blurRes: *Radio buttons, Default: Full.*
Selects the resolution factor for the motion blur. This is similar to the general 'Res' factor parameter, but does a better job of averaging down to lower resolution and interpolating back up to the result. Higher resolutions give better quality, lower resolutions give faster processing.

Full: Full resolution is used.
1/2: The motion blurring is performed at half resolution.
1/4: The motion blurring is performed at quarter resolution.

subpixel: *Toggle−button, Default:* on.
If enabled, uses a better quality but slightly slower method for performing the blur.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_BlurMoCurves(Source, 1.06, "Transform and Blur", shutterDuration, etc...);`

See Also:

BlurMotion
Blur
Streaks
WarpRepeat
WarpChroma
EdgeRays
Sapphire Plug−ins Introduction

BlurMotion

Performs a motion blur of the source clip between the specified From and To transformations. This can be used to perform radial zoom blurs, rotate blurs, directional blurs, or any combination of these. The From and To parameters do not refer to time. They describe the two transformations in space that determine the style of blur applied to each frame.

Inputs:

Source: The clip to be processed.

S_Mask: *Optional.* If connected, the amount of motion blur is scaled by this input for each destination pixel. Only the alpha channel of this input is used (or green/luma if there is no alpha).

Parameters:

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The center of rotation and zooming, in screen coordinates relative to the center of the frame. The shift values should be zero for this location to make sense.

exposureBias: *Default:* 0.5, *Range:* 0 to 1.
Determines the variable amount of exposure along the path between the From and To transformations. A value of 0 causes more exposure at the From end, 0.5 causes equal exposure along the path, and 1.0 causes more exposure at the To end. If you have bright spots on a dark background, a 0 value would cause the processed spots to be brighter at the From end and dark at the To end, and a 1.0 value would cause the opposite.

scaleResult: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

fromZDist: *Default:* 1, *Range:* 0.001 or greater.
The 'distance' of the From transformation. This zooms about the Center location when Shift is 0. Increase to zoom out, decrease to zoom in.

fromRotate: *Default:* 0, *Range:* any.
The rotation angle of the From transformation, in degrees, about the center.

fromShift: *X & Y, Default:* [0 0], *Range:* any.
The horizontal and vertical translations of the From transformation. This can be used for directional motion. If it is non−zero the center location becomes less meaningful.

toZDist: *Default:* 0.8, *Range:* 0.001 or greater.
The 'distance' of the To transformation. Increase to zoom out, or decrease to zoom in.

toRotate: *Default:* 0, *Range:* any.
The rotation angle of the To transformation, in degrees, about the center. Note that if the From and To Rotate angles are very different, the interpolation between them will become less accurate.

toShift: *X & Y, Default:* [0 0], *Range:* any.
The horizontal and vertical translations of the To transformation. This can be used for directional motion. If it is non−zero the center location becomes less meaningful.

wrap: *X & Y, Radio buttons, Default:* [*No No*].
Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.

Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

blurRes: *Radio buttons, Default: Full.*
Selects the resolution factor for the motion blur. This is similar to the general 'Res' factor parameter, but does a better job of averaging down to lower resolution and interpolating back up to the result. Higher resolutions give better quality, lower resolutions give faster processing.

Full: Full resolution is used.
1/2: The motion blurring is performed at half resolution.
1/4: The motion blurring is performed at quarter resolution.

subpixel: *Toggle−button, Default:* on.
If enabled, uses a better quality but slightly slower method for performing the blur.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_BlurMotion(Source, S_Mask, 1.06, centerX, etc...);`

See Also:

BlurMoCurves
Blur
Streaks
WarpRepeat
WarpChroma
EdgeRays
Sapphire Plug−ins Introduction

Clouds

In the S_Clouds Plugin.

Generates a procedural noise texture. Use the Frequency parameter to zoom in and out of the texture. The Shift Speed parameters cause the texture to automatically translate over time.

Inputs:

Background: The clip to combine the clouds image with. This may be ignored if the Combine option is set to Clouds Only, but the output clip will default to the same length as this clip.

Parameters:

frequency: *Default:* 2, *Range:* 0.01 or greater.
The spatial frequency of the clouds. Increase to zoom out, decrease to zoom in. Very high values of Frequency are clamped internally so the grain size is no smaller than a few pixels. If you want even finer grain use S_Grain or S_Clouds:Perspective instead.

frequencyRelX: *Default:* 0.4, *Range:* 0.01 or greater.
The relative horizontal frequency of the texture. Increase to stretch it vertically or decrease to stretch it horizontally.

octaves: *Integer, Default:* 8, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

seed: *Default:* 0.234, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

shiftStart: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translation offset of the texture. Since the texture is procedurally generated it can be shifted with no repeating units or seams occurring.

shiftSpeed: *X & Y, Default:* [.25*width 0], *Range:* any.
Translation speed of the texture. If non−zero, the result is automatically animated to shift at this rate. The result of animated Speed values may not be intuitive, so for variable speed motion it is usually best to set this to 0 and animate the Shift Start values instead.

brightness1: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of Color1. Increase for more contrast.

color1: *Default rgb:* [1 1 1].
The color of the 'brighter' parts of the texture. The colors of the result are determined by an interpolation between Color0 and Color1.

color0: *Default rgb:* [0 0 0].
The color of the 'darker' parts of the texture.

offset0: *Default:* 0, *Range:* any.
Adds this value to color0. Decrease to a negative value for more contrast.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
The background brightness is scaled by this value before being combined with the clouds.

combine: *Popup menu, Default: Screen.*
Determines how the texture is combined with the Background.

> ***Clouds Only:*** gives only the clouds texture with no Background.
> ***Mult:*** the texture is multiplied by the Background.
> ***Add:*** the texture is added to the Background.
> ***Screen:*** the texture is blended with the Background using a screen operation.
> ***Difference:*** the result is the difference between the texture and Background.
> ***Overlay:*** the texture is combined with the Background using an overlay function.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Clouds(Background, 1.06, "Clouds", frequency, etc...);`

See Also:

CloudsPerspective
CloudsVortex
CloudsMultColor
CloudsColorSmooth
CloudsPsyko
Grain
Sapphire Plug−ins Introduction

CloudsColorSmooth

In the S_Clouds Plugin.

Generates a full color clouds texture. Procedural noise texture is independently generated for each of the red, green, and blue output channels. The Shift Speed parameters cause the texture to automatically translate over time.

Inputs:

Background: The clip to combine the clouds image with. This may be ignored if the Combine option is set to Clouds Only, but the output clip will default to the same length as this clip.

Parameters:

frequency: *Default:* 8, *Range:* 0.01 or greater.
The spatial frequency of the clouds. Increase to zoom out, decrease to zoom in. Very high values of Frequency are clamped internally so the grain size is no smaller than a few pixels. If you want even finer grain use S_Grain or S_Clouds:Perspective instead.

frequencyRelX: *Default:* 0.2, *Range:* 0.01 or greater.
The relative horizontal frequency of the texture. Increase to stretch it vertically or decrease to stretch it horizontally.

octaves: *Integer, Default:* 1, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

seed: *Default:* 0.6, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

shiftStart: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translation offset of the texture. Since the texture is procedurally generated it can be shifted with no repeating units or seams occurring.

shiftSpeed: *X & Y, Default:* [.25*width 0], *Range:* any.
Translation speed of the texture. If non−zero, the result is automatically animated to shift at this rate. The result of animated Speed values may not be intuitive, so for variable speed motion it is usually best to set this to 0 and animate the Shift Start values instead.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

scaleColors: *Default rgb:* [1 1 1].
Scales the color of the result. For example, if it is yellow [1 1 0], the blue of the result will be 0.

saturation: *Default:* 1, *Range:* 0 or greater.
Scales the color saturation. Increase for more intense colors. Set to 0 for monochrome.

offset: *Default:* 0, *Range:* any.
Adds this gray value to the result (or subtracts if negative). 0 has no effect, .5 is middle gray, and 1 is white.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
The background brightness is scaled by this value before being combined with the clouds.

combine: *Popup menu, Default: Screen.*
Determines how the texture is combined with the Background.

Clouds Only: gives only the clouds texture with no Background.
Mult: the texture is multiplied by the Background.
Add: the texture is added to the Background.
Screen: the texture is blended with the Background using a screen operation.
Difference: the result is the difference between the texture and Background.
Overlay: the texture is combined with the Background using an overlay function.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Clouds(Background, 1.06, "Clouds Color Smooth", frequency, etc...);`

See Also:

Clouds
CloudsPerspective
CloudsVortex
CloudsMultColor
CloudsPsyko
Grain
Sapphire Plug−ins Introduction

CloudsMultColor

In the S_Clouds Plugin.

Generates a procedural noise texture like S_Clouds and tints the colors using an additional color noise texture. The Shift Speed parameters cause the texture to automatically translate over time.

Inputs:

Background: The clip to combine the clouds image with. This may be ignored if the Combine option is set to Clouds Only, but the output clip will default to the same length as this clip.

Parameters:

frequency: *Default:* 2, *Range:* 0.01 or greater.
The spatial frequency of the clouds. Increase to zoom out, decrease to zoom in. Very high values of Frequency are clamped internally so the grain size is no smaller than a few pixels. If you want even finer grain use S_Grain or S_Clouds:Perspective instead.

frequencyRelX: *Default:* 0.4, *Range:* 0.01 or greater.
The relative horizontal frequency of the texture. Increase to stretch it vertically or decrease to stretch it horizontally.

octaves: *Integer, Default:* 8, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

seed: *Default:* 0.234, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

colorAmount: *Default:* 0.5, *Range:* 0 or greater.
The amplitude of the color tinting.

colorFreq: *Default:* 1, *Range:* 0.01 or greater.
The frequency of the colors. Increase for finer color variation, decrease for softer color changes.

colorFreqRelx: *Default:* 0.4, *Range:* 0.01 or greater.
The relative horizontal frequency of the colors. Increase to stretch vertically, decrease to stretch horizontally.

colorOctaves: *Integer, Default:* 1, *Range:* 1 to 10.
The number of octaves of color noise to include. Each octave is twice the frequency and half the amplitude of the previous.

colorSeed: *Default:* 0.345, *Range:* 0 or greater.
The random number generator seed to use for the color noise. The actual seed value is not significant, but different values give different results.

shiftStart: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translation offset of the texture. Since the texture is procedurally generated it can be shifted with no repeating units or seams occurring.

shiftSpeed: *X & Y, Default:* [.25*width 0], *Range:* any.
Translation speed of the texture. If non−zero, the result is automatically animated to shift at this rate. The result of animated Speed values may not be intuitive, so for variable speed motion it is usually best to set this to 0 and animate the Shift Start values instead.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

color: *Default rgb:* [1 1 1].
Scales the color of the result.

offset: *Default:* 0, *Range:* any.
Adds this gray value to the result (or subtracts if negative). 0 has no effect, .5 is middle gray, and 1 is white.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
The background brightness is scaled by this value before being combined with the clouds.

combine: *Popup menu, Default: Screen.*
Determines how the texture is combined with the Background.

> ***Clouds Only:*** gives only the clouds texture with no Background.
> ***Mult:*** the texture is multiplied by the Background.
> ***Add:*** the texture is added to the Background.
> ***Screen:*** the texture is blended with the Background using a screen operation.
> ***Difference:*** the result is the difference between the texture and Background.
> ***Overlay:*** the texture is combined with the Background using an overlay function.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Clouds(Background, 1.06, "Clouds Mult Color", frequency, etc...);`

See Also:

Clouds
CloudsPerspective
CloudsVortex
CloudsColorSmooth
CloudsPsyko
Grain
Sapphire Plug−ins Introduction

CloudsPerspective

In the S_Clouds Plugin.

Generates a procedural noise texture transformed onto a 3D plane with perspective. Adjust the Latitude, Swing, and Roll parameters to rotate the image on various axes, each axis, and use the Frequency parameter to zoom in and out of the texture. Shift Speed causes the texture to automatically translate over time.

Inputs:

Background: The clip to combine the texture image with. This may be ignored if the Combine option is set to Texture Only, but the output clip will default to the same length as this clip.

Parameters:

frequency: *Default:* 2, *Range:* 0.01 or greater.
The spatial frequency of the texture. Increase to zoom out, decrease to zoom in.

frequencyRelX: *Default:* 0.4, *Range:* 0.01 or greater.
The relative horizontal frequency of the texture. Increase to stretch it vertically or decrease to stretch it horizontally.

octaves: *Integer, Default:* 6, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

seed: *Default:* 0.234, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the image. Values greater than 1.0 move it farther away and make it smaller. Values less then 1.0 move the image closer and enlarge it.

latitude: *Default:* −35, *Range:* −80 to 80.
Positive latitude tilts the image down and negative tilts it up. Keep latitude in the range of around −35 to 35 degrees to avoid aliasing towards the horizon.

swing: *Default:* 0, *Range:* any.
Rotation of the image in degrees in its initial frame.

roll: *Default:* 0, *Range:* any.
Tilts the result from side to side, in counter−clockwise degrees.

teleLensWidth: *Default:* 1, *Range:* 0.2 to 3.
The amount of lens telescoping. Increase to zoom in with less perspective, decrease for a wider viewing angle with more perspective.

shiftStart: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translation offset of the clouds in their initial plane.

shiftSpeed: *X & Y, Default:* [.25*width 0], *Range:* any.
Translation speed of the texture. If non−zero, the result is automatically animated to shift at this rate. The result of animated Speed values may not be intuitive, so for variable speed motion it is usually best to set this to 0 and animate the Shift Start values instead.

brightness1: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of Color1. Increase for more contrast.

color1: *Default rgb:* [1 1 1].
The color of the 'brighter' parts of the texture. The colors of the result are determined by an interpolation between Color0 and Color1.

color0: *Default rgb:* [0 0 0].
The color of the 'darker' parts of the texture.

offset0: *Default:* 0, *Range:* any.
Adds this value to color0. Decrease to a negative value for more contrast.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
The background brightness is scaled by this value before being combined with the texture.

combine: *Popup menu, Default: Screen.*
Determines how the texture is combined with the Background.

> ***Clouds Only:*** gives only the clouds texture with no Background.
> ***Mult:*** the texture is multiplied by the Background.
> ***Add:*** the texture is added to the Background.
> ***Screen:*** the texture is blended with the Background using a screen operation.
> ***Difference:*** the result is the difference between the texture and Background.
> ***Overlay:*** the texture is combined with the Background using an overlay function.

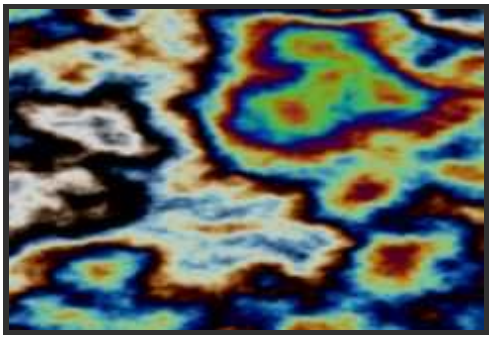
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Clouds(Background, 1.06, "Clouds Perspective", frequency, etc...);`

See Also:

Clouds
CloudsVortex
CloudsMultColor
CloudsColorSmooth
CloudsPsyko
WarpPerspective
Sapphire Plug−ins Introduction

CloudsPsyko



In the S_Clouds Plugin.

Generates a procedural noise texture, and passes this through a colorizing process. The Shift Speed parameters cause the pattern to automatically translate over time, and Phase Speed causes the colors to rotate over time.

Inputs:

Background: The clip to combine the clouds image with. This may be ignored if the Combine option is set to Clouds Only, but the output clip will default to the same length as this clip.

Parameters:

frequency: *Default:* 2, *Range:* 0.01 or greater.
The spatial frequency of the clouds. Increase to zoom out, decrease to zoom in. Very high values of Frequency are clamped internally so the grain size is no smaller than a few pixels. If you want even finer grain use S_Grain or S_Clouds:Perspective instead.

frequencyRelX: *Default:* 0.4, *Range:* 0.01 or greater.
The relative horizontal frequency of the texture. Increase to stretch it vertically or decrease to stretch it horizontally.

octaves: *Integer, Default:* 8, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

seed: *Default:* 0.23, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

colorFreq: *Default:* 4, *Range:* 0.01 or greater.
The frequency of the color pattern. Increase for a busier texture with more cycles through the spectrum.

freqRed: *Default:* 1, *Range:* 0 or greater.
The frequency of the red color component.

freqGreen: *Default:* 1.1, *Range:* 0 or greater.
The frequency of the green color component.

freqBlue: *Default:* 1.2, *Range:* 0 or greater.
The frequency of the blue color component.

shiftStart: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translation offset of the texture. Since the texture is procedurally generated it can be shifted with no repeating units or seams occurring.

shiftSpeed: *X & Y, Default:* [.25*width 0], *Range:* any.
Translation speed of the texture. If non−zero, the result is automatically animated to shift at this rate. The result of animated Speed values may not be intuitive, so for variable speed motion it is usually best to set this to 0 and animate the Shift Start values instead.

phaseStart: *Default:* −0.5, *Range:* any.
The phase offset of the color patterns.

phaseSpeed: *Default:* 0.3, *Range:* any.

The phase speed of the color patterns. If non−zero, the phase is automatically animated to give the color pattern a boiling look.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

scaleColor: *Default rgb:* [1 1 1].
Scales the color of the result. For example, if it is yellow [1 1 0], the blue of the result will be 0.

saturation: *Default:* 1, *Range:* 0 or greater.
Scales the strength of the colors. Increase for more intense colors, or decrease for muted colors.

offset: *Default:* 0, *Range:* any.
Adds this gray value to the result (or subtracts if negative). 0 has no effect, .5 is middle gray, and 1 is white.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
The background brightness is scaled by this value before being combined with the clouds.

combine: *Popup menu, Default: Screen.*
Determines how the texture is combined with the Background.

> ***Clouds Only:*** gives only the clouds texture with no Background.
> ***Mult:*** the texture is multiplied by the Background.
> ***Add:*** the texture is added to the Background.
> ***Screen:*** the texture is blended with the Background using a screen operation.
> ***Difference:*** the result is the difference between the texture and Background.
> ***Overlay:*** the texture is combined with the Background using an overlay function.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Clouds(Background, 1.06, "Clouds Psyko", frequency, etc...);`

See Also:

Clouds
CloudsPerspective
CloudsVortex
CloudsMultColor
CloudsColorSmooth
ZebrafyColor
PseudoColor
PsykoBlobs
PsykoStripes
Sapphire Plug−ins Introduction

CloudsVortex

In the S_Clouds Plugin.

Generates a procedural noise texture twisting into a vortex. The Vortex Speed parameter causes the amount of vortex rotation to automatically animate over time.

Inputs:

Background: The clip to combine the texture image with. This may be ignored if the Combine option is set to Texture Only, but the output clip will default to the same length as this clip.

Parameters:

frequency: *Default:* 2, *Range:* 0.01 or greater.
The spatial frequency of the texture. Increase to zoom out, decrease to zoom in.

frequencyRelX: *Default:* 1, *Range:* 0.01 or greater.
The relative horizontal frequency of the texture. Increase to stretch it vertically or decrease to stretch it horizontally.

octaves: *Integer, Default:* 6, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

seed: *Default:* 0.234, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The center of the vortex, in screen coordinates relative to the center of the frame.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the image. Values greater than 1.0 move it farther away and make it smaller. Values less then 1.0 move the image closer and enlarge it.

latitude: *Default:* 30, *Range:* −80 to 80.
Positive latitude tilts the image down and negative tilts it up. Keep latitude in the range of around −35 to 35 degrees to avoid aliasing towards the horizon.

vortexStart: *Default:* 72, *Range:* any.
The amount of vortex rotation, in approximate counter−clockwise degrees at the edge of the frame.

vortexSpeed: *Default:* 30, *Range:* any.
The speed of the vortex rotation, in approximate degrees per second at the edge of the frame. If non−zero, the vortexing is automatically animated at this rate.

angleOffset: *Default:* 0, *Range:* any.
If non−zero, a rotation is combined with the vortex. Make negative to rotate the inner and outer regions in opposite directions.

innerRadius: *Default:* .02*width, *Range:* 0 or greater.
The radius from the center at which the vortexing is phased in. This can be used to reduce excessive distortion and aliasing at the very center of the vortex.

brightness1: *Default:* 1, *Range:* 0 or greater.

Scales the brightness of Color1. Increase for more contrast.

color1: *Default rgb:* [1 1 1].
The color of the 'brighter' parts of the texture. The colors of the result are determined by an interpolation between Color0 and Color1.

color0: *Default rgb:* [0 0 0].
The color of the 'darker' parts of the texture.

offset0: *Default:* 0, *Range:* any.
Adds this value to color0. Decrease to a negative value for more contrast.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
The background brightness is scaled by this value before being combined with the texture.

combine: *Popup menu, Default: Screen.*
Determines how the texture is combined with the Background.

> ***Clouds Only:*** gives only the clouds texture with no Background.
> ***Mult:*** the texture is multiplied by the Background.
> ***Add:*** the texture is added to the Background.
> ***Screen:*** the texture is blended with the Background using a screen operation.
> ***Difference:*** the result is the difference between the texture and Background.
> ***Overlay:*** the texture is combined with the Background using an overlay function.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Clouds(Background, 1.06, "Clouds Vortex", frequency, etc...);`

See Also:

Clouds
CloudsPerspective
CloudsMultColor
CloudsColorSmooth
CloudsPsyko
WarpVortex
Sapphire Plug−ins Introduction

ColorOps: ClampChroma

In the S_ColorOps Plugin.

Reduces the chrominance of the input clip if necessary so it is not above a specified maximum. This effect can be used to make 'broadcast safe' colors. It also can be used to scale the chrominance, clamp the luminance, or scale the luminance.

Inputs:

Source: The clip to be processed.

Parameters:

clampChroma: *Default:* 0.5, *Range:* 0 to 1.
The maximum chrominance value. 1 is fully saturated and 0 is with no color. Source chrominance values below this will not be affected, but those above it will be reduced to it.

scaleChroma: *Default:* 1, *Range:* 0 or greater.
Scales the chrominance of all pixels. If this is 1 it will have no effect.

clampLuma: *Default:* 1, *Range:* 0 to 1.
The maximum luminance value. Source pixels brighter than this limit will be reduced to it. Values below it will not be affected. If this is 1 it will have no effect.

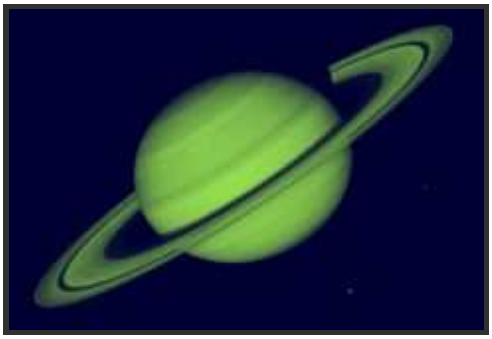
scaleLuma: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of all pixels. If this is 1 it will have no effect.

Script Form: `S_ColorOps(Source, 1.06, "Clamp Chroma", clampChroma, etc...);`

See Also:

HueSatBright
Monochrome
PseudoColor
DuoTone
Tint
Threshold
Hotspots
Gamma
Solarize
BlurChroma
Sapphire Plug−ins Introduction

ColorOps: DuoTone



In the S_ColorOps Plugin.

Performs an interpolation between two specified colors using the brightness of the source clip.

Inputs:

Source: The clip to be processed.

Parameters:

color1: *Default rgb:* [1 0.9 0.8].
The color to use at the brighter source regions.

color0: *Default rgb:* [0 0 0.2].
The color to use at the darker source regions.

invert: *Toggle−button, Default:* off.
If enabled, the resulting texture colors are inverted. This is similar to swapping Color0 and Color1.

threshold: *Default:* 0.5, *Range:* −0.5 to 1.5.
The source brightness value to use as the mid−point of the color interpolation. This is often a middle gray around 0.5.

softness: *Default:* 1, *Range:* 0.001 or greater.
The source brightness distance over which to perform the Color0 to Color1 interpolation. Decrease for sharper transitions between the two colors.

Script Form: `S_ColorOps(Source, 1.06, "Duo Tone", color1Red, etc...);`

See Also:

HueSatBright
Monochrome
ClampChroma
PseudoColor
Tint
Threshold
Hotspots
Gamma
Solarize

Sapphire Plug−ins Introduction

ColorOps: Gamma

In the S_ColorOps Plugin.

Applies a gamma correction to the input clip. The red, green, and blue channels can be adjusted independently. From Gamma just causes the inverse effect of adjusting Gamma.

Inputs:

Source: The clip to be processed.

Parameters:

gamma: *Default:* 1, *Range:* 0.1 or greater.
Values greater than 1.0 make the mid−tones brighter, values less than 1.0 make them darker, 1.0 leaves the input unchanged.

gammaRed: *Default:* 1, *Range:* 0.1 or greater.
Brightens or darkens the red mid−tones.

gammaGreen: *Default:* 1, *Range:* 0.1 or greater.
Brightens or darkens the green mid−tones.

gammaBlue: *Default:* 1, *Range:* 0.1 or greater.
Brightens or darkens the blue mid−tones.

fromGamma: *Default:* 1, *Range:* 0.1 or greater.
Divides the Gamma by this value before processing. This can be useful if your image was correct at this gamma, but needs to be adjusted from this to a new gamma.

fromGammaRed: *Default:* 1, *Range:* 0.1 or greater.
Darkens or brightens the red mid−tones.

fromGammaGreen: *Default:* 1, *Range:* 0.1 or greater.
Darkens or brightens the green mid−tones.

fromGammaBlue: *Default:* 1, *Range:* 0.1 or greater.
Darkens or brightens the blue mid−tones.

scaleLights: *Default:* 1, *Range:* 0 or greater.
Scales the brightness by this amount after the gamma correction. Increase for a brighter result.

offsetDarks: *Default:* 0, *Range:* any.
Adds this gray value to the darker regions after the gamma correction. This can be negative to increase contrast.

Script Form: `S_ColorOps(Source, 1.06, "Gamma", gamma, etc...);`

See Also:

HueSatBright
Monochrome
ClampChroma
PseudoColor
DuoTone

Sapphire Plug−ins Introduction

ColorOps: Hotspots

In the S_ColorOps Plugin.

Generates a hotspot image containing areas of the source clip brighter than a given threshold. The colors of the hotspots should match the original source. This can be used for increasing contrast or finding the bright areas of a clip, but without changing the color saturation or hue of the result.

Inputs:

Source: The clip to be processed.

Parameters:

blurInput: *Default:* 0, *Range:* 0 or greater.
Allows smaller spots to be smoothed away before the hotspots are determined.

threshold: *Default:* 0.7, *Range:* 0 to 1.
Include hotspots at any source areas that are brighter than this value.

thresholdAddColor: *Default rgb:* [0 0 0].
This can be used to raise the threshold on a specific color and thereby reduce the hotspots generated on areas of the source clip containing that color.

saturation: *Default:* 1, *Range:* any.
Scales the color saturation. Increase for more intense colors. Set to 0 for monochrome.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_ColorOps(Source, 1.06, "Hotspots", blurInput, etc...);`

See Also:

HueSatBright
Monochrome
ClampChroma
PseudoColor
DuoTone
Tint
Threshold
Gamma
Solarize

Sapphire Plug−ins Introduction

ColorOps: HueSatBright

In the S_ColorOps Plugin.

Adjusts the hue, saturation, brightness, and/or offset of the input clip.

Inputs:

Source: The clip to be processed.

Parameters:

hueShift: *Default:* 0, *Range:* −1 to 1.
Shifts the hue of the source colors, in revolutions from red to green to blue to red.

preserveLuma: *Toggle−button*, *Default:* off.
Enable this to preserve the brightness values of the input image after the hue is shifted.

saturation: *Default:* 1, *Range:* any.
Scales the color saturation of the result. Increase for more intense colors. Set to 0 for monochrome. You can also invert the chroma of the result by making this negative.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

offsetDarks: *Default:* 0, *Range:* any.
Adds this gray value to the darker regions of the result. This can be negative to increase contrast.

scaleColors: *Default rgb:* [1 1 1].
Scales the color of the result. For example, if it is yellow [1 1 0], the blue of the result will be 0.

Script Form: `S_ColorOps(Source, 1.06, "Hue Sat Bright", hueShift, etc...);`

See Also:

Monochrome
ClampChroma
PseudoColor
DuoTone
Tint
Threshold
Hotspots
Gamma
Solarize
Flicker
Sapphire Plug−ins Introduction

ColorOps: Monochrome

In the S_ColorOps Plugin.

Generates a monochrome version of the source clip using adjustable weights for the red, green, and blue channels. This can simulate the use of a color filter applied to the lens of a black and white camera. For example, use more red weight to darken blue sky areas of the input. The weights are scaled so they sum to 1 before being used to reduce overall brightness changes when they are adjusted.

Inputs:

Source: The clip to be processed.

Parameters:

weightRed: *Default:* 0.3, *Range:* −1 to 1.
The relative contribution of the source's red channel. To simulate a black and white exposure using a red filter, set this to 1 and set the green and blue weights to 0.

weightGreen: *Default:* 0.5, *Range:* −1 to 1.
The relative contribution of the source's green channel.

weightBlue: *Default:* 0.2, *Range:* −1 to 1.
The relative contribution of the source's blue channel

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

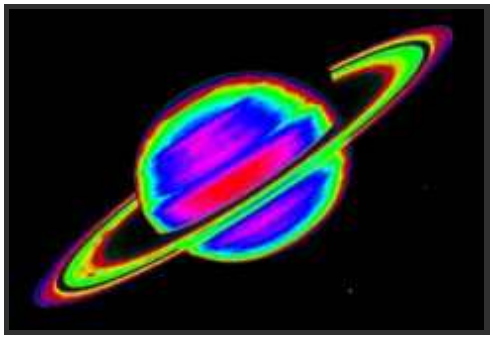
Script Form: `S_ColorOps(Source, 1.06, "Monochrome", weightRed, etc...);`

See Also:

HueSatBright
ClampChroma
PseudoColor
DuoTone
Tint
Threshold
Hotspots
Gamma
Solarize

Sapphire Plug−ins Introduction

ColorOps: PseudoColor



In the S_ColorOps Plugin.

Colorizes the source image. The hue is calculated from the brightness of the source.

Inputs:

Source: The clip to be processed.

Parameters:

frequency: *Default:* 2, *Range:* 0 or greater.
The frequency of the colorization. Increase for more cycles of hue through the spectrum, decrease for fewer.

hueShift: *Default:* 0, *Range:* −1 to 1.
Shift the color hues by this amount.

saturation: *Default:* 1, *Range:* any.
Scales the color saturation. Increase for more intense colors. Set to 0 for monochrome.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

scaleBySource: *Default:* 1, *Range:* 0 to 1.
The brightness of the output is scaled down by the original source brightness as this is increased to 1.

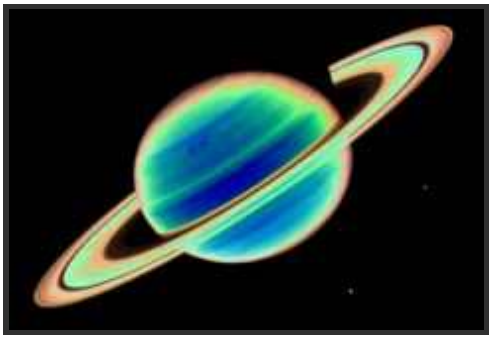
scaleBySrcAmp: *Default:* 1, *Range:* 0 or greater.
This amplifies the effect of Scale By Source, so if increased above 1, the middle grays can still retain their full brightness. It has no effect unless Scale By Source is positive.

Script Form: `S_ColorOps(Source, 1.06, "Pseudo Color", frequency, etc...);`

See Also:

HueSatBright
Monochrome
ClampChroma
DuoTone
Tint
Threshold
Hotspots
Gamma
Solarize
Zebrafy
ZebrafyColor
PsykoBlobs
PsykoStripes
Sapphire Plug−ins Introduction

ColorOps: Solarize



In the S_ColorOps Plugin.

Inverts the colors of the input clip that are brighter than the Threshold value, to create a 'solarization' effect.

Inputs:

Source: The clip to be processed.

Parameters:

threshold: *Default:* 0.5, *Range:* 0 to 1.
Colors above this value are inverted. If this is 0, all colors are inverted to produce a negative. If this is 1, no colors are inverted and the result should equal the input.

saturation: *Default:* 1, *Range:* any.
Scales the color saturation of the result. Increase for more intense colors. Set to 0 for monochrome. You can also invert the chroma of the result by making this negative.

invert: *Toggle−button, Default:* off.
If enabled, the result is inverted. The invert is applied before the Brightness and Offset are used, so you may need to readjust those parameters when you change the invert option.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result. Note that if a Threshold of .5 is used, no colors will be more than half the maximum brightness, so the contrast is increased by setting the Brightness to 2.

offsetDarks: *Default:* 0, *Range:* any.
Adds this gray value to the darker regions after the solarize effect. This can be negative to increase contrast.

Script Form: `S_ColorOps(Source, 1.06, "Solarize", threshold, etc...);`

See Also:

HueSatBright
Monochrome
ClampChroma
PseudoColor
DuoTone
Tint
Threshold
Hotspots
Gamma
ZebrafyColor
Sapphire Plug−ins Introduction

ColorOps: Threshold

In the S_ColorOps Plugin.

Sets the color channels of the source clip to full on or full off using a given softness and threshold. This can be used to increase the contrast of each color channel independently.

Inputs:

Source: The clip to be processed.

Parameters:

threshold: *Default:* 0.5, *Range:* 0 to 1.
The source brightness value to use as the mid−point of the thresholding. This is often a middle gray around .5.

thresholdAddColor: *Default rgb:* [0 0 0].
Raises the thresholds on each color channel using this color. It has no effect when black.

softness: *Default:* 0.2, *Range:* 0.001 to 1.
The softness of the transition between full off and on. Increase for smoother transitions, decrease for sharper ones.

softRelRed: *Default:* 1, *Range:* 0 or greater.
The relative softness of the red thresholding.

softRelGreen: *Default:* 1, *Range:* 0 or greater.
The relative softness of the red thresholding.

softRelBlue: *Default:* 1, *Range:* 0 or greater.
The relative softness of the red thresholding.

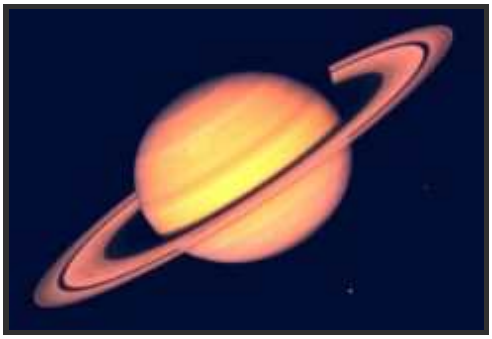
saturation: *Default:* 1, *Range:* any.
Scales the color saturation. Increase for more intense colors. Set to 0 for monochrome.

Script Form: `S_ColorOps(Source, 1.06, "Threshold", threshold, etc...);`

See Also:

HueSatBright
Monochrome
ClampChroma
PseudoColor
DuoTone
Tint
Hotspots
Gamma
Solarize
Sharpen
Sapphire Plug−ins Introduction

ColorOps: Tint



In the S_ColorOps Plugin.

Tints the dark and light regions of the input clip towards given colors. The dark colors are tinted by the Tint Dark color, and the brighter colors are tinted by the Tint Lights color.

Inputs:

Source: The clip to be processed.

Parameters:

tintLights: *Default rgb:* [1 1 1].
Scales the result by this color, thus tinting the lighter regions.

tintDarks: *Default rgb:* [0 0 0].
Adds this color to the darker regions of the source.

sourceSaturation: *Default:* 1, *Range:* any.
Scales the chroma saturation of the source. If this is zero you will see only color from the given tint colors.

scaleLights: *Default:* 1, *Range:* 0 or greater.
Scales the result by this gray value. Increase for a brighter result.

offsetDarks: *Default:* 0, *Range:* any.
Adds this gray value to the darker regions of the source. This can be negative to increase contrast.

Script Form: `S_ColorOps(Source, 1.06, "Tint", tintLightsRed, etc...);`

See Also:

HueSatBright
Monochrome
ClampChroma
PseudoColor
DuoTone
Threshold
Hotspots
Gamma
Solarize

Sapphire Plug−ins Introduction

Diffuse

Scrambles the pixels of the source input within an area determined by the Diffuse Amount. Use the Blur Rel X and Y parameters for a more horizontal or vertical diffuse direction. The pixelated look of this effect depends on the image resolution, so it is recommended to test your final resolution before processing.

Inputs:

Source: The clip to be processed.

S_Mask: *Optional.* If connected, determines which areas of the image receive diffusing pixels. Gray values internally scale the Diffuse Amount parameter rather than simply cross−fading between the effect and the original source. This can allow more continuous results at the mask edges and more detailed control over the diffusion amounts. The mask can be blurred using the Blur Mask parameter. Only the alpha channel of this input is used (or green/luma if there is no alpha).

Parameters:

diffuseAmount: *Default:* .05*width, *Range:* 0 or greater.
The amplitude of the pixel diffusion process.

relAmount: *X & Y, Default:* [1 1/GetDefaultAspect()], *Range:* 0 or greater.
Scales the relative horizontal and vertical amounts of diffusion.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the Mask input by this amount before using. This can be used to soften the edges of the mask and provide a smoother transition between the masked and unmasked areas. It has no effect unless the Mask input is connected.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Diffuse(Source, S_Mask, 1.06, diffuseAmount, etc...);`

See Also:

WipeDiffuse
DissolveDiffuse
Static
Grain
FilmEffect
Sapphire Plug−ins Introduction

DissolveBlur

In the S_Dissolves Plugin, and also in the Mixer menu of Shake's Transition node

Transitions between two inputs clips while blurring each. The first clip is blurred and faded out while the second clip is unblurred and faded in. The Dissolve Amount parameter should be animated to control the transition speed.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* 0 to 100.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

blurAmount: *Default:* .25*width, *Range:* 0 or greater.
Scales the width of the blur.

blurRel: *X & Y, Default:* [1 0], *Range:* 0 or greater.
The relative horizontal and vertical blur widths. Set Blur Rel X to 0 for a vertical−only blur, or set Blur Rel Y to 0 for a horizontal−only blur.

blurRelFrom: *Default:* 1, *Range:* 0 or greater.
Scales the amount of blur applied to the first clip. Set to 0 to fade out with no blur.

blurRelTo: *Default:* 1, *Range:* 0 or greater.
Scales the amount of blur applied to the second clip. Set to 0 to fade in with no blur.

blurFilter: *Popup menu, Default: Gauss.*
The type of convolution filter to blur with.

Box: uses a rectangular shaped filter.
Triangle: smoother, uses a pyramid shaped filter.
Gauss: smoothest, uses a gaussian shaped filter.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Dissolves(From, To, 1.06, "Dissolve Blur", mixPercent, etc...);`

See Also:

DissolveStatic
DissolveSpeckle
DissolveDiffuse
DissolveBubble
Blur
Sapphire Plug−ins Introduction

DissolveBubble

In the S_Dissolves Plugin, and also in the Mixer menu of Shake's Transition node

Transitions between two input clips using a bubble warping function. The first clip is warped away and faded out while the second clip is unwarped into place and faded in. The Dissolve Amount parameter should be animated to control the transition speed.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* 0 to 100.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

frequency: *Default:* 8, *Range:* 0.01 or greater.
The frequency of the bubble warping pattern. Increase for smaller bubbles, decrease for larger.

frequencyRelY: *Default:* GetDefaultAspect(), *Range:* 0.01 or greater.
The relative vertical frequency of the bubbles. Decrease for taller bubbles, increase for wider ones.

octaves: *Integer, Default:* 1, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

seed: *Default:* 0.23, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

amplitude: *Default:* 1, *Range:* any.
Scales the amount of warping distortion.

relAmp2: *Default:* −1, *Range:* any.
The relative amplitude of the second input clip warping distortion. If this is positive instead of negative, the clip will be unwarped from the opposite direction.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source images.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
The type of convolution filter to blur with.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Dissolves(From, To, 1.06, "Dissolve Bubble", mixPercent, etc...);`

See Also:

DissolveStatic
DissolveSpeckle
DissolveDiffuse
DissolveWaves
DissolvePuddle
DissolveVortex
DissolveBlur
DissolveLuma
WarpBubble
WipeBubble
Sapphire Plug−ins Introduction

DissolveDiffuse

In the S_Dissolves Plugin, and also in the Mixer menu of Shake's Transition node

Transitions between two input clips by scrambling the pixels of the inputs within an area determined by Max Amount. The first clip is diffused away while the second clip is diffused into place. The Dissolve Amount parameter should be animated to control the transition speed. The pixelated look of this effect depends on the image resolution, so it is recommended to test your final resolution before processing.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* 0 to 100.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

maxAmount: *Default:* .1*width, *Range:* 0 or greater.
Scales the magnitudes of the diffusion distances.

relAmount: *X & Y, Default:* [1 1/GetDefaultAspect()], *Range:* 0 or greater.
Scales the relative horizontal and vertical amounts of diffusion.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source images.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Dissolves(From, To, 1.06, "Dissolve Diffuse", mixPercent, etc...);`

See Also:

DissolveStatic
DissolveSpeckle
DissolveBubble
DissolveWaves
DissolvePuddle
DissolveVortex
DissolveBlur
DissolveLuma
Diffuse
WipeDiffuse
Sapphire Plug−ins Introduction

DissolveLuma

In the S_Dissolves Plugin, and also in the Mixer menu of Shake's Transition node

Transitions between two input clips using a pattern derived from their luminances. One clip often appears to emerge through the other. The Dissolve Amount parameter should be animated to control the transition speed.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* 0 to 100.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

softness: *Default:* 0.1, *Range:* 0 to 1.
Increase for softer and slower transitions.

useLumaOf: *Popup menu, Default: Difference.*
Determines how the transition pattern is generated from the clips' luminance values.

Difference: similar areas transition first, different areas last.
Subtract: areas where the first clip is brighter transition first, and areas where the second clip is brighter transition last.
Mult: areas where both images are bright transition first, and areas where either is dark are last.
Screen: areas where either image is bright transition first, and areas where both are dark transition last.
From: dark areas of the first clip disappear first, bright areas last.
To: bright areas of the second clip appear first, dark areas last.

invertPattern: *Toggle−button, Default:* off.
If enabled, the transition pattern is reversed in time.

smoothPattern: *Default:* 0, *Range:* 0 or greater.
If positive, a blur is applied to the transition pattern. This can reduce noise and give clearer edges to transition lines.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Dissolves(From, To, 1.06, "Dissolve Luma", mixPercent, etc...);`

See Also:

DissolveStatic
DissolveSpeckle
DissolveDiffuse
DissolveBubble

Sapphire Plug−ins Introduction

DissolvePuddle

In the S_Dissolves Plugin, and also in the Mixer menu of Shake's Transition node

Transitions between two input clips while warping by a circular pattern of waves. The first clip is warped away and faded out while the second clip is unwarped into place and faded in. The Dissolve Amount parameter should be animated to control the transition speed.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* 0 to 100.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

frequency: *Default:* 5, *Range:* 0.01 or greater.
The frequency of the puddle pattern. Increase for more and smaller puddle, or decrease for fewer and larger.

relHeight: *Default:* 0.75, *Range:* 0.01 or greater.
The relative height of the concentric wave pattern.

center: *X & Y*, *Default:* [.5*width .5*height], *Range:* any.
The location of the puddle center in screen coordinates relative to the center of the frame. This parameter can be set by enabling and moving the Center Widget. Note that moving the puddle center can also cause the puddle size to change so that the current value of Wipe Amt remains correct.

amplitude: *Default:* 0.2, *Range:* any.
Scales the amount of warping distortion.

relAmp2: *Default:* −1, *Range:* any.
The relative amplitude of the second input clip warping distortion. If this is positive instead of negative, the clip will be unwarped from the opposite direction.

rotatePuddle: *Default:* 0, *Range:* any.
Rotates the puddle pattern by this many counter−clockwise degrees after the Rel Height stretching has been applied. This has no effect when Rel Height is 1.

phaseStart: *Default:* 0, *Range:* any.
The phase shift of the waves.

phaseSpeed: *Default:* 1, *Range:* any.
The speed of the waves. If this is positive the waves automatically travel outwards from the center at this rate.

innerRadius: *Default:* 0, *Range:* any.
The distance from the puddle center where the wave distortion is phased in. No waves are generated inside this radius.

innerSoftness: *Default:* .05*width, *Range:* 2 or greater.
The width of the region at the Inner Radius over which the wave distortion is phased in.

outerRadius: *Default:* .75*width, *Range:* 0 or greater.
The distance from the puddle center where the wave distortion is phased out. No waves are generated outside this radius.

outerSoftness: *Default:* .2*width, *Range:* 2 or greater.
The width of the region at the Outer Radius over which the wave distortion is phased out.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source images.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Dissolves(From, To, 1.06, "Dissolve Puddle", mixPercent, etc...);`

See Also:

DissolveStatic
DissolveSpeckle
DissolveDiffuse
DissolveBubble
DissolveWaves
DissolveVortex
DissolveBlur
DissolveLuma
WarpPuddle
Sapphire Plug−ins Introduction

DissolveSpeckle

In the S_Dissolves Plugin, and also in the Mixer menu of Shake's Transition node

Transition between two input clips using a speckled noise pattern. The Dissolve Amount parameter should be animated to control the transition speed.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* 0 to 100.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

frequency: *Default:* 40, *Range:* 0.01 or greater.
The frequency of the speckle pattern. Increase for smaller speckles, decrease for larger.

frequencyRelY: *Default:* GetDefaultAspect(), *Range:* 0.01 or greater.
The relative vertical frequency of the speckles pattern. Increase for wider speckles, decrease for taller speckles.

octaves: *Integer*, *Default:* 1, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

seed: *Default:* 0.23, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Dissolves(From, To, 1.06, "Dissolve Speckle", mixPercent, etc...);`

See Also:

DissolveStatic
DissolveDiffuse
DissolveBubble
DissolveWaves
DissolvePuddle
DissolveVortex
DissolveBlur
DissolveLuma

Sapphire Plug−ins Introduction

DissolveStatic

In the S_Dissolves Plugin, and also in the Mixer menu of Shake's Transition node

Transitions between two input clips using random pixel static. The Dissolve Amount parameter should be animated to control the transition speed. The pixelated look of this effect depends on the image resolution, so it is recommended to test your final resolution before processing.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* 0 to 100.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Dissolves(From, To, 1.06, "Dissolve Static", mixPercent, etc...);`

See Also:

DissolveSpeckle
DissolveDiffuse
DissolveBubble
DissolveWaves
DissolvePuddle
DissolveVortex
DissolveBlur
DissolveLuma
Static
FilmEffect
Sapphire Plug−ins Introduction

DissolveVortex

In the S_Dissolves Plugin, and also in the Mixer menu of Shake's Transition node

Transitions between two input clips using a vortex warping function. The first clip is warped away and faded out while the second clip is unwarped into place and faded in. The Dissolve Amount parameter should be animated to control the transition speed.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* 0 to 100.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The location of the vortex center in screen coordinates relative to the center of the frame. This parameter can be set by enabling and moving the Center Widget. Note that moving the vortex center can also cause the vortex size to change so that the current value of Wipe Amt remains correct.

vortexAmount: *Default:* 72, *Range:* any.
The amount of vortex rotation, in approximate degrees at the edge of the frame.

relAmount2: *Default:* −1, *Range:* any.
The relative amount of the second clip vortex rotation. If this is positive instead of negative the second clip will be unvortexed from the opposite direction.

rotateAmount: *Default:* 0, *Range:* any.
If non−zero, a rotation is also added to the warping. Make negative to rotate the inner and outer regions in different directions.

innerRadius: *Default:* .02*width, *Range:* 0 or greater.
The radius from the center at which the vortexing is phased in. This can be used to reduce excessive distortion and aliasing at the very center of the vortex.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source images.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Dissolves(From, To, 1.06, "Dissolve Vortex", mixPercent, etc...);`

See Also:

DissolveStatic
DissolveSpeckle
DissolveDiffuse
DissolveBubble
DissolveWaves
DissolvePuddle
DissolveBlur
DissolveLuma
WarpVortex
Sapphire Plug−ins Introduction

DissolveWaves

In the S_Dissolves Plugin, and also in the Mixer menu of Shake's Transition node

Transitions between two input clips using a waves warping function. The first clip is warped away and faded out while the second clip is unwarped into place and faded in. The Dissolve Amount parameter should be animated to control the transition speed.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* 0 to 100.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

frequency: *Default:* 3, *Range:* 0.01 or greater.
The frequency of the waves pattern. Increase for more and smaller waves, or decrease for fewer and larger.

amplitude: *Default:* 0.3, *Range:* any.
Scales the amount of warping distortion.

relAmp2: *Default:* −1, *Range:* any.
The relative amplitude of the second input clip warping distortion. If this is positive instead of negative, the clip will be unwarped from the opposite direction.

angle: *Default:* 45, *Range:* any.
The rotation of the overall waves pattern used for the wipe, in counter−clockwise degrees.

displaceAngle: *Default:* 90, *Range:* any.
The warping direction in degrees relative to the angle of the waves. 0 gives compression−expansion waves, and 90 gives side to side waves.

phaseStart: *Default:* 0, *Range:* any.
The phase shift of the waves. The wave pattern is translated in the direction of Angle by this amount.

phaseSpeed: *Default:* 0, *Range:* any.
The phase speed of the waves. If this is non−zero the wave pattern automatically travels at this rate.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source images.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Dissolves(From, To, 1.06, "Dissolve Waves", mixPercent, etc...);`

See Also:

DissolveStatic
DissolveSpeckle
DissolveDiffuse
DissolveBubble
DissolvePuddle
DissolveVortex
DissolveBlur
DissolveLuma
WarpWaves
Sapphire Plug−ins Introduction

Distort

In the S_Distort Plugin.

Warps the source input clip using the gradient of the Lens input clip. This can generate optical glass−like effects as if the source clip were being viewed through an arbitrarily shaped lens. It is best demonstrated when the lens image contains just a few bold shapes or a simple texture.

Inputs:

Source: The clip to be processed.

Lens: Distorts the source using the brightness values of this input clip.

S_Mask: *Optional.* If connected, the amount of lens distortion is scaled by this input, so the Source is unaffected where the mask is black. This input can be affected using the blurMask or invertMask parameters. Only the alpha channel of this input is used (or green/luma if there is no alpha).

Parameters:

amount: *Default:* 1, *Range:* any.
The severity of the lens distortions. Make negative to invert the direction of the distortions.

fine: *Toggle−button, Default:* off.
If enabled, the warping amount is reduced by a factor of 100. This mode is meant to allow subtle expansion or contraction of the source image near the edges of its matte given as the Lens input.

blurLens: *Default:* .05*width, *Range:* 0 or greater.
Smooths out the edges in the lens image by this amount before using it.

rotateWarpDir: *Default:* 0, *Range:* any.
Rotates the warping direction by this many degrees. If non−zero, this can add some unusual twisting effects to the lens distortion.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Distort(Source, Lens, S_Mask, 1.06, "Distort", amount, etc...);`

See Also:

DistortBlur
DistortChroma
DistortRGB
EmbossDistort
WarpFishEye
Sapphire Plug−ins Introduction

DistortBlur

In the S_Distort Plugin.

Blurs the source input clip in the direction of the gradient of the Lens input clip. It is best demonstrated when the lens image contains just a few simple shapes.

Inputs:

Source: The clip to be processed.

Lens: Distorts the source using the brightness values of this input clip.

S_Mask: *Optional.* If connected, the amount of lens distortion is scaled by this input, so the Source is unaffected where the mask is black. This input can be affected using the blurMask or invertMask parameters. Only the alpha channel of this input is used (or green/luma if there is no alpha).

Parameters:

blurAmount: *Default:* 1, *Range:* 0 or greater.
The magnitude of the blur distortions.

warpAmount: *Default:* 0, *Range:* any.
Adds some additional non−blurred lens distortion if non−zero.

blurLens: *Default:* .05*width, *Range:* 0 or greater.
Smooths out the edges in the lens image by this amount before using it.

rotateBlurDir: *Default:* 0, *Range:* any.
Rotates the blurring direction by this many degrees. If non−zero, this can add some unusual twisting effects to the blurring.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

subpixel: *Toggle−button, Default:* on.
If enabled, uses a better quality but slightly slower method for performing the blur.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Distort(Source, Lens, S_Mask, 1.06, "Distort Blur", blurAmount, etc...);`

See Also:

Distort
DistortChroma
DistortRGB
Blur
BlurMotion
Sapphire Plug−ins Introduction

DistortChroma

In the S_Distort Plugin.

Warps the chrominance of the source input by different amounts using the gradient of the Lens input clip. This can generate optical glass−like effects as if the source clip were being viewed through an arbitrarily shaped or textured prism. It is best demonstrated when the lens image contains just a few simple bold shapes.

Inputs:

Source: The clip to be processed.

Lens: Distorts the source using the brightness values of this input clip.

S_Mask: *Optional.* If connected, the amount of lens distortion is scaled by this input, so the Source is unaffected where the mask is black. This input can be affected using the blurMask or invertMask parameters. Only the alpha channel of this input is used (or green/luma if there is no alpha).

Parameters:

amount: *Default:* 1, *Range:* any.
The severity of the lens distortions. Make negative to invert the direction of the distortions.

blurLens: *Default:* .05*width, *Range:* 0 or greater.
Smooths out the edges in the lens image by this amount before using it.

rotateWarpDir: *Default:* 0, *Range:* any.
Rotates the warping direction by this many degrees. If non−zero, this can add some unusual twisting effects to the lens distortion.

warpRed: *Default:* 0.5, *Range:* any.
The magnitude of lens distortion for the red end of the spectrum. Make negative to invert the direction of the red distortions.

warpBlue: *Default:* 1, *Range:* any.
The magnitude of lens distortion for the blue end of the spectrum. Make negative to invert the direction of the blue distortions.

steps: *Integer, Default:* 8, *Range:* 3 to 100.
The number of color samples along the spectrum to include. More steps give a smoother result, but require more time to process.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Distort(Source, Lens, S_Mask, 1.06, "Distort Chroma", amount, etc...);`

See Also:

Distort
DistortBlur
DistortRGB
EmbossGlass
WarpChroma
Sapphire Plug−ins Introduction

DistortRGB

In the S_Distort Plugin.

Warps the red, green, and blue color channels of the source input by different amounts using the gradient of the Lens input clip. It is best demonstrated when the lens image contains just a few simple bold shapes.

Inputs:

Source: The clip to be processed.

Lens: Distorts the source using the brightness values of this input clip.

S_Mask: *Optional.* If connected, the amount of lens distortion is scaled by this input, so the Source is unaffected where the mask is black. This input can be affected using the blurMask or invertMask parameters. Only the alpha channel of this input is used (or green/luma if there is no alpha).

Parameters:

amount: *Default:* 1, *Range:* any.
Scales the magnitude of the lens distortion for all channels. Make negative to invert the direction of the distortions.

blurLens: *Default:* .05*width, *Range:* 0 or greater.
Smooths out the edges in the lens image by this amount before using it.

rotateWarpDir: *Default:* 0, *Range:* any.
Rotates the warping direction by this many degrees. If non−zero, this can add some unusual twisting effects to the lens distortion.

warpRed: *Default:* 0.5, *Range:* any.
Scales the amount of lens distortion for the red channel. Negate to invert the direction.

warpGreen: *Default:* 0.75, *Range:* any.
Scales the amount of lens distortion for the red channel. Negate to invert the direction.

warpBlue: *Default:* 1, *Range:* any.
Scales the amount of lens distortion for the red channel. Negate to invert the direction.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Distort(Source, Lens, S_Mask, 1.06, "Distort RGB", amount, etc...);`

See Also:

Distort
DistortBlur
DistortChroma
EmbossGlass
WarpChroma
Sapphire Plug−ins Introduction

DropShadow

Generates a shadow on the Background clip using the alpha channel of the Foreground, then composites the Foreground over the Background to give the final result.

Inputs:

Foreground: The clip to use as foreground, and the alpha channel of this clip is used as the matte to generate the shadow.

Background: *Optional.* The shadow is drawn onto this Background clip.

Matte: *Optional.* If this is provided, its alpha channel is used instead of the Foreground to generate the shadow. This input can be affected by the Invert Matte or Matte Use parameters.

Parameters:

shadowColor: *Default rgb:* [0 0 0].
The color of the shadow.

shadowOpacity: *Default:* 1, *Range:* 0 or greater.
The opacity of the shadow, use values near 0 for subtle transparent shadows, or values near 1.0 for stronger shadows.

shadowBlur: *Default:* .01*width, *Range:* 0 or greater.
Determines the softness of the shadow.

shift_: *X & Y, Default:* [.02*width −.02*width], *Range:* any.
The horizontal and vertical offset of the shadow.

fgOpacity: *Default:* 1, *Range:* 0 to 1.
Scales the opacity of the Foreground without affecting the shadow. Lowering this can be used to fade out the Foreground, or setting it to zero prevents the Foreground from being composited over the result at all.

compPremult: *Toggle−button, Default:* on.
Enable this for a better composite if the Foreground pixel colors have been pre−multiplied by their Matte values. This is also known as an 'additive' composite.

invertMatte: *Toggle−button, Default:* off.
If enabled, the black and white of the Foreground alpha channel are inverted before use.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_DropShadow(Foreground, Background, Matte, 1.06, shadowColorRed, etc...);`

See Also:

Blur
Sapphire Plug−ins Introduction

EdgeBlur

Finds the edges within the Matte clip, and blurs the Source clip at those edges. Use the Show Edges option to view which areas will receive the blur while adjusting the edge parameters. Then adjust Blur Width to control the amount of blur.

Inputs:

Source: The clip to be processed.

Matte: *Optional.* The clip used to determine the edge locations where the Source should be blurred. Only the brightness value of this clip is used. If this input is not connected, the alpha of the Input clip is used instead to determine the edges.

Parameters:

blurWidth: *Default:* 10, *Range:* 0 or greater.
The width of the blur. This should normally not be much greater than the Edge Width.

edgeWidth: *Default:* 10, *Range:* 0 or greater.
The width of the edge area to blur within.

edgeStrength: *Default:* 0.5, *Range:* 0 or greater.
The strength of the edges determines the amount of the blurred source that replaces the edges.

edgeThreshold: *Default:* 0, *Range:* 0 or greater.
Determines which edges are blurred. Increase to remove minor edges or speckles.

show: *Radio buttons, Default: Result.*
Selects between output options.

Result: outputs the Source image with blurred edges.
Edges: outputs only the edge image. This can useful during the adjustment of the edge parameters.

subpixel: *Toggle−button, Default:* off.
Enables blurring by subpixel amounts. Use this for smoother animation of the Blur Width or Edge Width parameters.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_EdgeBlur(Source, Matte, 1.06, blurWidth, etc...);`

See Also:

Blur
EdgeFlash
EdgeDetect
Sapphire Plug−ins Introduction

EdgeDetect

In the S_EdgeDetect Plugin.

Finds the edges within the source clip. Increase the Edge Smooth parameter for thicker edges. Select Mono or Chroma mode to show only edges in Luminance or Chroma.

Inputs:

Source: The clip to be processed.

Parameters:

edgeSmooth: *Default:* 0, *Range:* 0 or greater.
Increase for thicker and smoother edges.

subpixelSmooth: *Toggle−button*, *Default:* off.
Enables smoothing the edges by subpixel amounts. Use this for smoother animation of the Edge Smooth parameter.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

saturation: *Default:* 1, *Range:* 0 or greater.
Scales the color saturation of the result. Increase for more intense colors. Set to 0 for monochrome.

threshold: *Default:* 0, *Range:* 0 or greater.
Subtracts this value from the result. Increase to remove unwanted noise from minor edges.

weightRed: *Default:* 1, *Range:* 0 or greater.
Scale the edges of the red source channel.

weightGreen: *Default:* 1, *Range:* 0 or greater.
Scale the edges of the green source channel.

weightBlue: *Default:* 1, *Range:* 0 or greater.
Scale the edges of the blue source channel.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_EdgeDetect(Source, 1.06, "Edge Detect", edgeSmooth, etc...);`

See Also:

EdgeDetectMono
EdgeDetectChroma
EdgesInDirection
EdgeDetectDouble
EdgeColorize
BandPass
Sharpen
Emboss
Sapphire Plug−ins Introduction

EdgeDetect: Colorize

In the S_EdgeDetect Plugin.

Assigns different colors to the edges of the source clip depending on their direction. Increase the Edge Smooth parameter for thicker edges.

Inputs:

Source: The clip to be processed.

Parameters:

edgeSmooth: *Default:* 0, *Range:* 0 or greater.
Increase for thicker and smoother edges.

subpixelSmooth: *Toggle−button*, *Default:* off.
Enables smoothing the edges by subpixel amounts. Use this for smoother animation of the Edge Smooth parameter.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the edge colors by this amount.

rotateColors: *Default:* 0, *Range:* any.
Causes the Top, Left, Right, and Bottom colors to be rotated to different edge directions, in counter−clockwise degrees.

background: *Default rgb:* [0 0 0].
The color to use as a background.

top: *Default rgb:* [1 0.85 0.5].
The color of upwards facing edges.

right: *Default rgb:* [0 0.1 0.5].
The color of right facing edges.

bottom: *Default rgb:* [0.3 0.3 0.3].
The color of downwards facing edges.

left: *Default rgb:* [0.5 0 0].
The color of left facing edges.

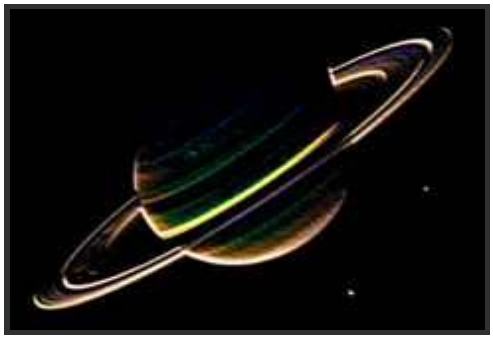
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_EdgeDetect(Source, 1.06, "Edge Colorize", edgeSmooth, etc...);`

See Also:

EdgeDetect
EdgeDetectMono
EdgeDetectChroma
EdgesInDirection
EdgeDetectDouble
BandPass
Sharpen
Emboss
Sapphire Plug−ins Introduction

EdgeDetect: InDirection



In the S_EdgeDetect Plugin.

Finds the edges of the source input that are aligned in a specified direction. Increase the Edge Smooth parameter for thicker edges.

Inputs:

Source: The clip to be processed.

Parameters:

edgeSmooth: *Default:* 0, *Range:* 0 or greater.
Increase for thicker and smoother edges.

subpixelSmooth: *Toggle−button*, *Default:* off.
Enables smoothing the edges by subpixel amounts. Use this for smoother animation of the Edge Smooth parameter.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

saturation: *Default:* 1, *Range:* 0 or greater.
Scales the color saturation of the result. Increase for more intense colors. Set to 0 for monochrome.

offsetColor: *Default rgb:* [0 0 0].
The color to add to the result. Make this gray to allow the darker side of edges away from the given Direction to also be visible.

direction: *X & Y*, *Default:* [.75*width .75*height], *Range:* any.
Edges are found which are perpendicular to this direction. To allow visual adjustment with the crosshairs widget, the direction vector used is from the center of the screen to these pixel coordinates.

bidirectional: *Toggle−button*, *Default:* off.
If enabled, edges towards and away from the Direction vector are treated equally.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_EdgeDetect(Source, 1.06, "Edges In Direction", edgeSmooth, etc...);`

See Also:

EdgeDetect
EdgeDetectMono
EdgeDetectChroma
EdgeDetectDouble
EdgeColorize
BandPass
Sharpen
Emboss
Sapphire Plug−ins Introduction

EdgeDetectChroma

In the S_EdgeDetect Plugin.

Similar to EdgeDetect, but luminance edges are ignored, and only the edges in the Source's chrominance are found. This option can sometimes be helpful for use with matte extraction.

Inputs:

Source: The clip to be processed.

Parameters:

edgeSmooth: *Default:* 0, *Range:* 0 or greater.
Increase for thicker and smoother edges.

subpixelSmooth: *Toggle−button*, *Default:* off.
Enables smoothing the edges by subpixel amounts. Use this for smoother animation of the Edge Smooth parameter.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

saturation: *Default:* 1, *Range:* 0 or greater.
Scales the color saturation of the result. Increase for more intense colors. Set to 0 for monochrome.

threshold: *Default:* 0, *Range:* 0 or greater.
Subtracts this value from the result. Increase to remove unwanted noise from minor edges.

weightRed: *Default:* 1, *Range:* 0 or greater.
Scale the edges of the red source channel.

weightGreen: *Default:* 1, *Range:* 0 or greater.
Scale the edges of the green source channel.

weightBlue: *Default:* 1, *Range:* 0 or greater.
Scale the edges of the blue source channel.

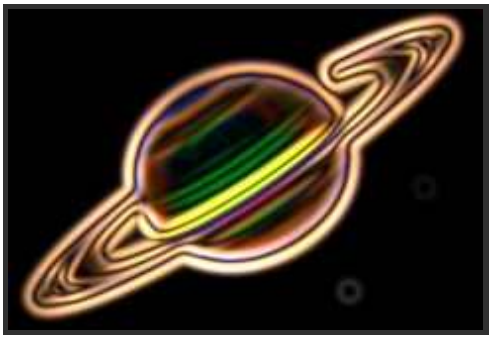
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_EdgeDetect(Source, 1.06, "Edge Detect Chroma", edgeSmooth, etc...);`

See Also:

EdgeDetect
EdgeDetectMono
EdgesInDirection
EdgeDetectDouble
EdgeColorize
BandPass
Sharpen
Emboss
Sapphire Plug−ins Introduction

EdgeDetectDouble



In the S_EdgeDetect Plugin.

Performs an edge detect operation twice giving a double stranded edge effect. Increase the Edge Smooth parameters for thicker edges.

Inputs:

Source: The clip to be processed.

Parameters:

edgeSmooth1: *Default:* .01*width, *Range:* 0 or greater.
Increase for smoother edges for the first edge−detect.

edgeSmooth2: *Default:* 0, *Range:* 0 or greater.
Increase for smoother edges for the second edge−detect.

subpixelSmooth: *Toggle−button, Default:* off.
Enables smoothing the edges by subpixel amounts. Use this for smoother animation of the Edge Smooth parameter.

brightness1: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the initial edges.

brightness2: *Default:* 1.5, *Range:* 0 or greater.
Scales the brightness of the result.

saturation: *Default:* 1, *Range:* 0 or greater.
Scales the color saturation of the result. Increase for more intense colors. Set to 0 for monochrome.

threshold1: *Default:* 0, *Range:* 0 or greater.
Subtract this value from the initial edges.

threshold2: *Default:* 0, *Range:* 0 or greater.
Subtracts this value from the result. Increase to remove unwanted noise from minor edges.

show: *Radio buttons, Default: Edges2.*
Selects the output option.

> ***Edges1:*** shows just the first edge−detect. This can be useful for adjusting the Edge1 parameters without performing the second edge−detect.
> ***Edges2:*** shows the result of the double edge−detect.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_EdgeDetect(Source, 1.06, "Edge Detect Double", edgeSmooth1, etc...);`

See Also:

EdgeDetect
EdgeDetectMono
EdgeDetectChroma
BandPass
Sharpen
Sapphire Plug−ins Introduction

EdgeDetectMono

In the S_EdgeDetect Plugin.

Similar to EdgeDetect, but first makes the source monochrome, and then finds the edges within the resulting single channel (faster).

Inputs:

Source: The clip to be processed.

Parameters:

edgeSmooth: *Default:* 0, *Range:* 0 or greater.
Increase for thicker and smoother edges.

subpixelSmooth: *Toggle−button*, *Default:* off.
Enables smoothing the edges by subpixel amounts. Use this for smoother animation of the Edge Smooth parameter.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

threshold: *Default:* 0, *Range:* 0 or greater.
Subtracts this value from the result. Increase to remove unwanted noise from minor edges.

weightRed: *Default:* 1, *Range:* 0 or greater.
Scale the edges of the red source channel.

weightGreen: *Default:* 1, *Range:* 0 or greater.
Scale the edges of the green source channel.

weightBlue: *Default:* 1, *Range:* 0 or greater.
Scale the edges of the blue source channel.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_EdgeDetect(Source, 1.06, "Edge Detect Mono", edgeSmooth, etc...);`

See Also:

EdgeDetect
EdgeDetectChroma
EdgesInDirection
EdgeDetectDouble
EdgeColorize
BandPass
Sharpen
Emboss
Sapphire Plug−ins Introduction

EdgeFlash

Adds a glow from the Front clip onto the Back clip, and vice versa, then composites the Front over the Back. This can be used to make a composite look more natural with light flashing between the layers as if exposed on film together.

Inputs:

Foreground: The clip to use as foreground.

Background: The clip to use as background.

Matte: *Optional.* Specifies the opacities of the Foreground clip. Only the alpha channel of this input is used (or green/luma if there is no alpha). If this input is not connected, the alpha channel of the Foreground input is used instead.

Parameters:

fgFlashAmp: *Default:* 0.8, *Range:* 0 or greater.
The amount of flashing from the Front onto the Back.

bgFlashAmp: *Default:* 0.8, *Range:* 0 or greater.
The amount of flashing from the Front onto the Back.

flashWidth: *Default:* .01*width, *Range:* 0 or greater.
The width of the flashing.

output: *Radio buttons, Default: Comp.*
Selects between different output options.

Foreground: outputs only the Front clip with flashing from the Back.
Background: outputs only the Back clip with flashing from the Front.
Comp: flashes both, composites the Front over the Back, and outputs the result.

subpixelWidths: *Toggle−button, Default:* off.
Enables flashing by subpixel amounts. Use this for smoother animation of the flash width.

compPremult: *Toggle−button, Default:* on.
Enable this for a better composite if the Front pixel values have been pre−multiplied by the Matte pixel values. This is also known as an 'additive' composite.

invertMatte: *Toggle−button, Default:* off.
If enabled, the black and white of the matte are inverted before use.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

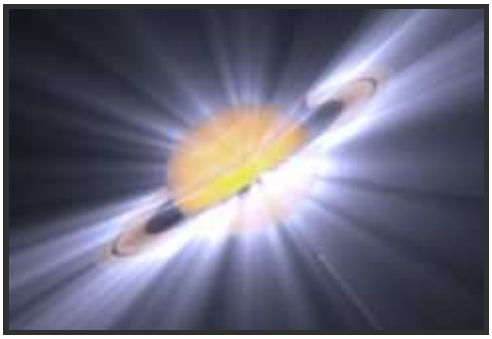
clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_EdgeFlash(Foreground, Background, Matte, 1.06, fgFlashAmp, etc...);`

See Also:

Blur
EdgeBlur
EdgeDetect
Glow
Sapphire Plug−ins Introduction

EdgeRays



Generates beams of light emitting from the edges of an input clip.

Inputs:

Source: The clip to be processed.

Background: *Optional.* The clip to use as background.

S_Mask: *Optional.* If connected, the ray colors are scaled by this input. A monochrome Mask can be used to choose a subset of areas that will generate rays. If the Mask Type is set to Color, a color Mask input can be used to selectively adjust the ray colors in different regions. This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

raysLength: *Default:* 0.25, *Range:* 1 or less.
The length of the rays. A length of 1.0 actually gives rays that continue forever, although though they may still fade out as they go. To make the rays look longer you can also increase the Rel Outer Bright parameter. If Rays Length is negative the rays can beam inwards instead of outwards. Note that processing times increase for longer rays.

raysColor: *Default rgb:* [1 1 1].
Scales the color of the ray beams.

raysBrightness: *Default:* 2, *Range:* 0 or greater.
Scales the brightness of the ray beams.

biasOuterBright: *Default:* 0, *Range:* 0 to 1.
Determines the variable amount of brightness along the rays. This is normally near 0 so the rays fade away at their outer ends, 0.5 causes equal brightness along the rays, and 1.0 causes maximum brightness at the ends.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The location from which the rays beam outwards.

raysRes: *Radio buttons, Default: Full.*
Selects the resolution factor for the rays. Higher resolutions give sharper rays, lower resolutions give smoother rays and faster processing. This 'Res' factor only affects the rays: the background is still combined with the rays at full resolution.

Full: Full resolution is used.
1/2: The rays are calculated at half resolution.
1/4: The rays are calculated at quarter resolution.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the rays. The maximum of the red, green, and blue ray brightness is scaled by this value and combined with the background Alpha at each pixel.

show: *Radio buttons, Default: Result.*
Selects between output options.

Result: outputs the rays over the Background.
Edges: outputs only the edge image. This can useful during the adjustment of the edge or shimmer parameters.

edgeThickness: *Default:* .005*width, *Range:* 0 or greater.
The thickness of the edges which generate the rays.

edgeBrightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the edges which generate the rays.

edgeSubpixel: *Toggle−button, Default:* off.
Enables subpixel Edge Thickness amounts. Turn this on you are animating Edge Thickness or if you want finer control of small values.

shimmerAmp: *Default:* 0.5, *Range:* 0 or greater.
Modulates the ray source image with this amount of grain texture to give the rays a shimmering look.

shimmerFreq: *Default:* 40, *Range:* 0.1 or greater.
The frequency of the shimmer texture. Increase for a finer grained shimmer effect, decrease for softer grain. This has no effect unless Shimmer Amp is positive.

shimmerSeed: *Default:* 0.123, *Range:* 0 or greater.
Used to initialize the random number generator for the shimmer texture. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

shimmerShift: *X & Y, Default:* [0 0], *Range:* any.
Translation of the shimmer texture. This has no effect unless Shimmer Amp is positive.

shimmerSpeed: *X & Y, Default:* [0 0], *Range:* any.
Translation speed of the shimmer texture. If non−zero, the shimmering is automatically animated to shift at this rate.

raysFromAlpha: *Default:* 0, *Range:* 0 to 1.
Set to 1 to generate rays from the edges of the source's alpha channel instead of its RGB channels. This will typically reduce the rays generated from internal edges. Values between 0 and 1 interpolate between using the RGB and the Alpha.

raysUnderSource: *Default:* 0, *Range:* 0 to 1.
Set to 1 to composite the Source input over the rays.

scaleSource: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the Source input when combined with the rays. This does not affect the generation of the rays themselves.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background. This parameter only has an effect if the background input is provided.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

maskType: *Radio buttons, Default: Mono.*
This setting is ignored unless the Mask input is connected.

> ***Mono:*** uses the luminance of the Mask input to scale the brightness of the rays.
> ***Color:*** uses the RGB channels of the Mask input to scale the colors of the rays.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked

and unmasked areas. This has no effect unless the S_Mask input is connected.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_EdgeRays(Source, Background, S_Mask, 1.06, raysLength, etc...);`

See Also:

Rays
Streaks
BlurMotion
WarpChroma
EdgeDetect
Glow
Sapphire Plug−ins Introduction

Emboss

In the S_Emboss Plugin.

Embosses the Source clip using the brightness of the Bumps input as a relief map. Increase the Bumps Smooth parameter for bolder bumps, and adjust the Light Dir to illuminate the bumps from different angles.

Inputs:

Source: The clip to be processed.

Bumps: The bump map for the emboss. Only the luminance of this input is used.

S_Mask: *Optional.* If connected, the emboss is applied only at the areas specified by this input. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

lightDir: *X & Y, Default:* [.25*width .75*height], *Range:* any.
The direction vector for the light source. Surface shading is calculated using light from this direction shining onto the Bumps input.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

lightColor: *Default rgb:* [1 1 1].
The color of the light source that creates the embossed result.

bumpsScale: *Default:* 1, *Range:* any.
Scales the amplitude of the bump map.

bumpsThreshold: *Default:* 0, *Range:* 0 to 1.
This value is subtracted from the Bumps input before it is used.

bumpsSmooth: *Default:* 0, *Range:* 0 or greater.
If positive, the Bumps input is blurred by this amount before being used. Increase for a softer emboss effect.

subpixelSmooth: *Toggle−button, Default:* off.
If enabled, the amount of pre−smoothing of the Bumps input is performed at subpixel accuracy. It can be helpful if Bumps Smooth is very small or is being animated. This parameter has no effect unless Bumps Smooth is positive.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Emboss(Source, Bumps, S_Mask, 1.06, "Emboss", lightDirX, etc...);`

See Also:

EmbossShiny
EmbossDistort
EmbossGlass
Distort
EdgesInDirection
Sapphire Plug−ins Introduction

EmbossDistort

In the S_Emboss Plugin.

Embosses and warps the Source clip using the Bumps input as a relief map and also distorts the result using the Bumps as a 'lens' image. Increase the Bumps Smooth parameter for bolder bumps, and adjust the Light Dir to illuminate the bumps from different angles.

Inputs:

Source: The clip to be processed.

Bumps: The bump map and lens source for the emboss.

S_Mask: *Optional.* If connected, the emboss is applied only at the areas specified by this input. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

lightDir: *X & Y, Default:* [.25*width .75*height], *Range:* any.
The direction vector for the light source. Surface shading is calculated using light from this direction shining onto the Bumps input.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

lightColor: *Default rgb:* [1 1 1].
The color of the light source that creates the embossed result.

bumpsScale: *Default:* 1, *Range:* any.
Scales the amplitude of the bump map.

bumpsThreshold: *Default:* 0, *Range:* 0 to 1.
This value is subtracted from the Bumps input before it is used.

bumpsSmooth: *Default:* .01*width, *Range:* 0 or greater.
If positive, the Bumps input is blurred by this amount before being used. Increase for a softer emboss effect.

subpixelSmooth: *Toggle−button, Default:* off.
If enabled, the amount of pre−smoothing of the Bumps input is performed at subpixel accuracy. It can be helpful if Bumps Smooth is very small or is being animated. This parameter has no effect unless Bumps Smooth is positive.

hilightBrightness: *Default:* 0.5, *Range:* 0 to 1.
Scales the brightness of the specular highlights.

hilightSize: *Default:* 0.5, *Range:* 0.1 or greater.
Adjusts the size of the specular highlights.

distortAmount: *Default:* 1, *Range:* any.
The severity of the lens warping distortion. Make negative to invert the direction of the distortions.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the

image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

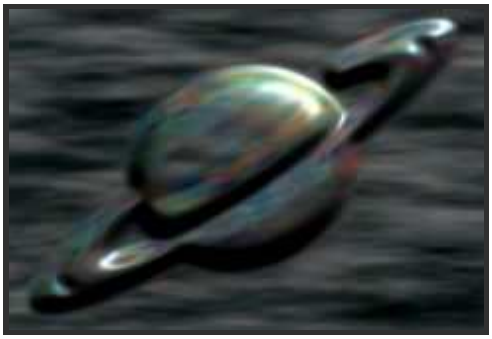
invertMask: *Toggle−button*, *Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Emboss(Source, Bumps, S_Mask, 1.06, "Emboss Distort", lightDirX, etc...);`

See Also:

Emboss
EmbossShiny
EmbossGlass
Distort
Sapphire Plug−ins Introduction

EmbossGlass



In the S_Emboss Plugin.

The Source is embossed and warped using the Bumps input as a relief map and lens image. A chrominance distortion is also performed, separating the spectrum for a 'prismatic' look. Increase the Bumps Smooth parameter for bolder bumps, and adjust the Light Dir to illuminate the bumps from different angles.

Inputs:

Source: The clip to be processed.

Bumps: The bump map and lens source for the emboss.

S_Mask: *Optional.* If connected, the emboss is applied only at the areas specified by this input. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

lightDir: *X & Y, Default:* [.25*width .75*height], *Range:* any.
The direction vector for the light source. Surface shading is calculated using light from this direction shining onto the Bumps input.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

lightColor: *Default rgb:* [1 1 1].
The color of the light source that creates the embossed result.

bumpsScale: *Default:* 1, *Range:* any.
Scales the amplitude of the bump map.

bumpsThreshold: *Default:* 0, *Range:* 0 to 1.
This value is subtracted from the Bumps input before it is used.

bumpsSmooth: *Default:* .01*width, *Range:* 0 or greater.
If positive, the Bumps input is blurred by this amount before being used. Increase for a softer emboss effect.

subpixelSmooth: *Toggle−button, Default:* off.
If enabled, the amount of pre−smoothing of the Bumps input is performed at subpixel accuracy. It can be helpful if Bumps Smooth is very small or is being animated. This parameter has no effect unless Bumps Smooth is positive.

hilightBrightness: *Default:* 0.5, *Range:* 0 to 1.
Scales the brightness of the specular highlights.

hilightSize: *Default:* 0.5, *Range:* 0.1 or greater.
Adjusts the size of the specular highlights.

distortAmount: *Default:* 1, *Range:* any.
The severity of the lens warping distortion. Make negative to invert the direction of the distortions.

filter: *Toggle−button, Default:* on.

If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

warpRed: *Default:* 0.5, *Range:* any.
The magnitude of lens distortion for the red end of the spectrum. Make negative to invert the direction of the red distortions.

warpBlue: *Default:* 1, *Range:* any.
The magnitude of lens distortion for the blue end of the spectrum. Make negative to invert the direction of the blue distortions.

steps: *Integer*, *Default:* 5, *Range:* 3 to 100.
The number of color samples along the spectrum to include. More steps give a smoother result, but require more time to process.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button*, *Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Emboss(Source, Bumps, S_Mask, 1.06, "Emboss Glass", lightDirX, etc...);`

See Also:

Emboss
EmbossShiny
EmbossDistort
DistortChroma
Sapphire Plug−ins Introduction

EmbossShiny

In the S_Emboss Plugin.

Embosses the Source clip using the Bumps input as a relief map. A lighting model is used which includes highlights from specular reflections. Increase the Bumps Smooth parameter for bolder bumps, and adjust the Light Dir to illuminate the bumps from different angles.

Inputs:

Source: The clip to be processed.

Bumps: The bump map for the emboss. Only the luminance of this input is used.

S_Mask: *Optional.* If connected, the emboss is applied only at the areas specified by this input. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

lightDir: *X & Y, Default:* [.25*width .75*height], *Range:* any.
The direction vector for the light source. Surface shading is calculated using light from this direction shining onto the Bumps input.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

lightColor: *Default rgb:* [1 1 1].
The color of the light source that creates the embossed result.

bumpsScale: *Default:* 1, *Range:* any.
Scales the amplitude of the bump map.

bumpsThreshold: *Default:* 0, *Range:* 0 to 1.
This value is subtracted from the Bumps input before it is used.

bumpsSmooth: *Default:* .002*width, *Range:* 0 or greater.
If positive, the Bumps input is blurred by this amount before being used. Increase for a softer emboss effect.

subpixelSmooth: *Toggle−button, Default:* off.
If enabled, the amount of pre−smoothing of the Bumps input is performed at subpixel accuracy. It can be helpful if Bumps Smooth is very small or is being animated. This parameter has no effect unless Bumps Smooth is positive.

hilightBrightness: *Default:* 0.8, *Range:* 0 to 1.
Scales the brightness of the specular highlights.

hilightSize: *Default:* 0.5, *Range:* 0.1 or greater.
Adjusts the size of the specular highlights.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Emboss(Source, Bumps, S_Mask, 1.06, "Emboss Shiny", lightDirX, etc...);`

See Also:

Emboss
EmbossDistort
EmbossGlass
TextureNoiseEmboss
Sapphire Plug−ins Introduction

Feedback

In the S_Feedback Plugin.

The previous frames of the input clip are transformed and combined with the current frame to give a variety of effects inspired by video feedback. The output of each processed frame is stored and then combined with the next frame. The feedback is reinitialized whenever any non−consecutive frame is processed: either the first frame, reprocessing a given frame, or jumping to another frame. You must process multiple frames of a clip in a row to observe the effect.

Inputs:

Source: The clip to be processed.

Parameters:

prevColor: *Default rgb:* [1 1 1].
For each frame, the previous output is scaled by this color before it is combined with the new input frame. This is similar to Prev Brightness but affects the colors of the previous frames instead of just the brightness.

prevBrightness: *Default:* 0.8, *Range:* 0 or greater.
For each frame, the previous output is scaled by this amount before it is combined with the new input frame. Normally this value should be less than 1.0 which causes previous frames to fade out over time. A value of 1.0 causes no fading, and values greater than 1.0 cause previous frames to become brighter over time.

prevHueShift: *Default:* 0, *Range:* any.
Shifts the hue of the previous frames' colors, for each new frame.

newColor: *Default rgb:* [1 1 1].
Scales the color of the current frame. Set this to the complement of Old Color to offset overly colored trails.

newOpacity: *Default:* 1, *Range:* 0 or greater.
Scales the opacity and brightness of the current frame.

combineNew: *Popup menu, Default: Ave.*
Selects the method for combining previous frames with the current frame.

Ave: The current frame is averaged with the previous output, smearing moving objects out over time. The previous frames are scaled by oldBrightness and the input is scaled by 1.0−oldBrightness for a weighted average, so oldBrightness must be less than 1.0 for this to work properly. Unlike the other combine options, Ave should never affect the brightness of stationary objects in the clip.

Max: The colors of the current frame and previous frames are combined with a maximum function. This makes the output frame at least as bright as the current frame, and will make brighter 'trails' for example if you have bright objects moving on a dark background.

Screen: The colors of the current frame and previous frames are combined with a blend function. This can be used to accumulate the colors of a moving clip. However, non−black regions will become brighter with each frame.

Add: The colors of the current frame and previous frames are added. This can also be used to accumulate the colors of a moving clip, with the non−black regions becoming brighter at each frame.

Over: The current frame is composited over the previous frames using its Alpha channel. This uses

pre−multiplied compositing, so where the alpha is black the Source image should normally also be black. If the input clip contains no Alpha channel, the luminance is used instead.

Under: The current frame is composited under the previous frames.

blurAmount: *Default:* 0, *Range:* 0 or greater.
The previous frames are blurred by this amount for each new frame. This has no effect unless it is positive.

diffuseAmount: *Default:* 0, *Range:* 0 or greater.
The previous frames are passed through a pixel−diffusion process of this magnitude, for each new frame. This has no effect unless it is positive.

amountRel: *X & Y, Default:* [1 1], *Range:* 0 or greater.
The relative amounts of horizontal and vertical blurring and/or diffusing. This has no effect unless Blur Amount or Diffuse Amount are positive.

zDist: *Default:* 0.95, *Range:* 0.001 or greater.
For each new frame, the 'distance' of the previous frames is scaled by this amount. This causes zooming during the feedback process. Values greater than 1.0 zoom out and make the previous frames smaller, and values less then 1.0 zoom in and enlarge them.

rotate: *Default:* 3, *Range:* any.
For each new frame, the amount of rotation in degrees to apply to the previous frames.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The center position for rotation and scaling, in screen coordinates relative to the center of the frame.

shift_: *X & Y, Default:* [0 0], *Range:* any.
Shifts the previous frames by this amount for each new frame. If this is non−zero the Center location is less meaningful.

wrap: *X & Y, Radio buttons, Default:* [*No No*].
Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Feedback(Source, 1.06, "Feedback", prevColorRed, etc...);`

See Also:

FeedbackBubble
Trails
TrailsDiffuse
NearestColor
WarpRepeat
WarpChroma
Sapphire Plug−ins Introduction

Feedback: NearestColor

In the S_Feedback Plugin.

Collects pixel colors from the input clip's frames that are closest to the given Match Color. This can create, for example, a background−only image from a clip with objects moving over a blue or green−screen background. It can also be used to accumulate the color of a moving object over a non−colored background. The collected colors are reinitialized whenever any non−consecutive frame is processed, either the first frame, reprocessing a given frame, or jumping to another frame. You must process multiple frames of a clip in a row to observe the effect.

Inputs:

Source: The clip to be processed.

Parameters:

matchColor: *Default rgb:* [0 0 1].
Pixel colors are kept that are 'nearest' to this color.

chromaWeight: *Default:* 1, *Range:* 0 or greater.
The amount of influence hue has on the color matching. If this is 0, the pixels with the closest brightness to Match Color will be kept; if it is 2, the hue will have more influence and the brightness will have less.

Script Form: `S_Feedback(Source, 1.06, "Nearest Color", matchColorRed, etc...);`

See Also:

Feedback
FeedbackBubble
Trails
TrailsDiffuse
WarpRepeat
WarpChroma
Sapphire Plug−ins Introduction

Feedback: Trails

In the S_Feedback Plugin.

The previous frames of the input clip are combined with the current frame to give a variety of 'time trails' effects. The output of each processed frame is stored and then combined with the next frame. The trails are reinitialized whenever any non−consecutive frame is processed, either the first frame, reprocessing a given frame, or jumping to another frame. You must process multiple frames of a clip in a row to observe the effect.

Inputs:

Source: The clip to be processed.

Parameters:

prevColor: *Default rgb:* [1 1 1].
For each frame, the previous output is scaled by this color before it is combined with the new input frame. This is similar to Prev Brightness but affects the colors of the previous frames instead of just the brightness.

prevBrightness: *Default:* 0.8, *Range:* 0 or greater.
For each frame, the previous output is scaled by this amount before it is combined with the new input frame. Normally this value should be less than 1.0 which causes previous frames to fade out over time. A value of 1.0 causes no fading, and values greater than 1.0 cause previous frames to become brighter over time.

prevHueShift: *Default:* 0, *Range:* any.
Shifts the hue of the previous frames' colors, for each new frame.

newColor: *Default rgb:* [1 1 1].
Scales the color of the current frame. Set this to the complement of Old Color to offset overly colored trails.

newOpacity: *Default:* 1, *Range:* 0 or greater.
Scales the opacity and brightness of the current frame.

combineNew: *Popup menu, Default: Ave.*
Selects the method for combining previous frames with the current frame.

Ave: The current frame is averaged with the previous output, smearing moving objects out over time. The previous frames are scaled by oldBrightness and the input is scaled by 1.0−oldBrightness for a weighted average, so oldBrightness must be less than 1.0 for this to work properly. Unlike the other combine options, Ave should never affect the brightness of stationary objects in the clip.

Max: The colors of the current frame and previous frames are combined with a maximum function. This makes the output frame at least as bright as the current frame, and will make brighter 'trails' for example if you have bright objects moving on a dark background.

Screen: The colors of the current frame and previous frames are combined with a blend function. This can be used to accumulate the colors of a moving clip. However, non−black regions will become brighter with each frame.

Add: The colors of the current frame and previous frames are added. This can also be used to accumulate the colors of a moving clip, with the non−black regions becoming brighter at each frame.

Over: The current frame is composited over the previous frames using its Alpha channel. This uses pre−multiplied compositing, so where the alpha is black the Source image should normally also be black.

If the input clip contains no Alpha channel, the luminance is used instead.

Under: The current frame is composited under the previous frames.

blurAmount: *Default:* 0, *Range:* 0 or greater.
The previous frames are blurred by this amount for each new frame. This has no effect unless it is positive.

blurRel: *X & Y*, *Default:* [1 1/GetDefaultAspect()], *Range:* 0 or greater.
The relative horizontal and vertical amounts of the blurring. This has no effect unless Blur Amount is positive.

Script Form: `S_Feedback(Source, 1.06, "Trails", prevColorRed, etc...);`

See Also:

Feedback
FeedbackBubble
TrailsDiffuse
NearestColor
WarpRepeat
WarpChroma
Sapphire Plug−ins Introduction

Feedback: TrailsDiffuse

In the S_Feedback Plugin.

The previous frames of the input clip are processed with a pixel diffusion process and then combined with the current frame. The output of each processed frame is stored and then combined with the next frame. The trails are reinitialized whenever a non−consecutive frame is processed, either the first frame, reprocessing a given frame, or jumping to another frame. You must process multiple frames of a clip in a row to observe the effect.

Inputs:

Source: The clip to be processed.

Parameters:

prevColor: *Default rgb:* [1 1 1].
For each frame, the previous output is scaled by this color before it is combined with the new input frame. This is similar to Prev Brightness but affects the colors of the previous frames instead of just the brightness.

prevBrightness: *Default:* 0.8, *Range:* 0 or greater.
For each frame, the previous output is scaled by this amount before it is combined with the new input frame. Normally this value should be less than 1.0 which causes previous frames to fade out over time. A value of 1.0 causes no fading, and values greater than 1.0 cause previous frames to become brighter over time.

prevHueShift: *Default:* 0, *Range:* any.
Shifts the hue of the previous frames' colors, for each new frame.

newColor: *Default rgb:* [1 1 1].
Scales the color of the current frame. Set this to the complement of Old Color to offset overly colored trails.

newOpacity: *Default:* 1, *Range:* 0 or greater.
Scales the opacity and brightness of the current frame.

combineNew: *Popup menu, Default: Ave.*
Selects the method for combining previous frames with the current frame.

Ave: The current frame is averaged with the previous output, smearing moving objects out over time. The previous frames are scaled by oldBrightness and the input is scaled by 1.0−oldBrightness for a weighted average, so oldBrightness must be less than 1.0 for this to work properly. Unlike the other combine options, Ave should never affect the brightness of stationary objects in the clip.

Max: The colors of the current frame and previous frames are combined with a maximum function. This makes the output frame at least as bright as the current frame, and will make brighter 'trails' for example if you have bright objects moving on a dark background.

Screen: The colors of the current frame and previous frames are combined with a blend function. This can be used to accumulate the colors of a moving clip. However, non−black regions will become brighter with each frame.

Add: The colors of the current frame and previous frames are added. This can also be used to accumulate the colors of a moving clip, with the non−black regions becoming brighter at each frame.

Over: The current frame is composited over the previous frames using its Alpha channel. This uses pre−multiplied compositing, so where the alpha is black the Source image should normally also be black.

If the input clip contains no Alpha channel, the luminance is used instead.

Under: The current frame is composited under the previous frames.

diffuseAmount: *Default:* .01*width, *Range:* 0 or greater.
The previous frames are passed through a pixel−diffusion process of this magnitude, for each new frame. This has no effect unless it is positive.

diffuseRel: *X & Y, Default:* [1 1/GetDefaultAspect()], *Range:* 0 or greater.
The relative horizontal and vertical amounts of the pixel diffusion process. This has no effect unless Diffuse Amount is positive.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

Script Form: `S_Feedback(Source, 1.06, "Trails Diffuse", prevColorRed, etc...);`

See Also:

Feedback
FeedbackBubble
Trails
NearestColor
WarpRepeat
WarpChroma
Sapphire Plug−ins Introduction

FeedbackBubble

In the S_Feedback Plugin.

The previous frames of the input clip are distorted by a solid noise pattern, transformed, and combined with the current frame to give a variety of possible effects. The output of each processed frame is stored and then combined with the next frame. The feedback is reinitialized whenever any non−consecutive frame is processed, either the first frame, reprocessing a given frame, or jumping to another frame. You must process multiple frames of a clip in a row to observe the effect.

Inputs:

Source: The clip to be processed.

Parameters:

prevColor: *Default rgb:* [1 1 1].
For each frame, the previous output is scaled by this color before it is combined with the new input frame. This is similar to Prev Brightness but affects the colors of the previous frames instead of just the brightness.

prevBrightness: *Default:* 0.8, *Range:* 0 or greater.
For each frame, the previous output is scaled by this amount before it is combined with the new input frame. Normally this value should be less than 1.0 which causes previous frames to fade out over time. A value of 1.0 causes no fading, and values greater than 1.0 cause previous frames to become brighter over time.

prevHueShift: *Default:* 0, *Range:* any.
Shifts the hue of the previous frames' colors, for each new frame.

newColor: *Default rgb:* [1 1 1].
Scales the color of the current frame. Set this to the complement of Old Color to offset overly colored trails.

newOpacity: *Default:* 1, *Range:* 0 or greater.
Scales the opacity and brightness of the current frame.

combineNew: *Popup menu, Default: Ave.*
Selects the method for combining previous frames with the current frame.

Ave: The current frame is averaged with the previous output, smearing moving objects out over time. The previous frames are scaled by oldBrightness and the input is scaled by 1.0−oldBrightness for a weighted average, so oldBrightness must be less than 1.0 for this to work properly. Unlike the other combine options, Ave should never affect the brightness of stationary objects in the clip.

Max: The colors of the current frame and previous frames are combined with a maximum function. This makes the output frame at least as bright as the current frame, and will make brighter 'trails' for example if you have bright objects moving on a dark background.

Screen: The colors of the current frame and previous frames are combined with a blend function. This can be used to accumulate the colors of a moving clip. However, non−black regions will become brighter with each frame.

Add: The colors of the current frame and previous frames are added. This can also be used to accumulate the colors of a moving clip, with the non−black regions becoming brighter at each frame.

Over: The current frame is composited over the previous frames using its Alpha channel. This uses

pre−multiplied compositing, so where the alpha is black the Source image should normally also be black. If the input clip contains no Alpha channel, the luminance is used instead.

Under: The current frame is composited under the previous frames.

bubbleAmount: *Default:* 0.05, *Range:* any.
The amplitude of the noise pattern used to create the distortion.

bubbleFreq: *Default:* 16, *Range:* 0.01 or greater.
The spatial frequency of the initial noise pattern. Increase to zoom out, decrease to zoom in.

bubbleFreqRelX: *Default:* 1, *Range:* 0.01 or greater.
The relative horizontal frequency of the noise pattern. Increase to stretch it vertically or decrease to stretch it horizontally.

bubbleShift: *X & Y, Default:* [0 0], *Range:* −360 or greater.
The horizontal and vertical translation of the noise pattern.

bubbleShiftSpeed: *X & Y, Default:* [0 .05*height], *Range:* −360 or greater.
If non−zero, the bubble pattern is automatically animated to shift at this speed.

bubbleOctaves: *Integer, Default:* 1, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

bubbleSeed: *Default:* 0.123, *Range:* 0 or greater.
Used to initialize the random number generator for the noise pattern. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

blurAmount: *Default:* 0, *Range:* 0 or greater.
The previous frames are blurred by this amount for each new frame. This has no effect unless it is positive.

diffuseAmount: *Default:* 0, *Range:* 0 or greater.
The previous frames are passed through a pixel−diffusion process of this magnitude, for each new frame. This has no effect unless it is positive.

amountRel: *X & Y, Default:* [1 1], *Range:* 0 or greater.
The relative amounts of horizontal and vertical blurring and/or diffusing. This has no effect unless Blur Amount or Diffuse Amount are positive.

zDist: *Default:* 0.95, *Range:* 0.001 or greater.
For each new frame, the 'distance' of the previous frames is scaled by this amount. This causes zooming during the feedback process. Values greater than 1.0 zoom out and make the previous frames smaller, and values less then 1.0 zoom in and enlarge them.

rotate: *Default:* 0, *Range:* any.
For each new frame, the amount of rotation in degrees to apply to the previous frames.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The center position for rotation and scaling, in screen coordinates relative to the center of the frame.

shift_: *X & Y, Default:* [0 0], *Range:* any.
Shifts the previous frames by this amount for each new frame. If this is non−zero the Center location is less meaningful.

wrap: *X & Y, Radio buttons, Default:* [*No No*].
Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Feedback(Source, 1.06, "FeedbackBubble", prevColorRed, etc...);`

See Also:

Feedback
Trails
TrailsDiffuse
NearestColor
WarpRepeat
WarpChroma
Sapphire Plug−ins Introduction

FilmEffect

Provides a physically accurate model of film exposure and processing to make your video footage look like it was shot on film. It can remove field artifacts, perform color correction for specific film types, add film grain, and apply glow or soft focus effects. The color correction and grain can be selectively disabled using the Scale CC and Grain Amp parameters.

Inputs:

Source: The clip to be processed.

Parameters:

negFilm: *Popup menu, Default: Kodak_5245.*
Selects the negative film stock.

None: Ignore any effect of negative film. This is not normally useful unless you also select None for the print film to disable both.
Kodak_5245: Eastman EXR 50D, low speed, daylight balanced, very fine grain.
Kodak_5246: Kodak VISION 250D, higher contrast, medium speed, daylight balanced, fine grain.
Kodak_5248: Eastman EXR 100T, medium speed, tungsten light balanced, very fine grain.
Kodak_5274: Kodak VISION 200T, medium speed, tungsten light balanced, fine grain.
Kodak_5277: Kodak VISION 320T, lower contrast, medium speed, tungsten light balanced, medium−fine grain.
Kodak_5279: Kodak VISION 500T, high speed, tungsten light balanced, somewhat grainy.
Kodak_5284: Kodak VISION Expression 500T, lower contrast, high speed, tungsten light balanced, medium grain.
Kodak_5289: Kodak VISION 800T, very fast, tungsten light balanced, grainy.
Kodak_5293: Eastman EXR 200T, reduced contrast, tungsten light balanced, medium grain.
Kodak_5298: Eastman EXR 500T, high speed, tungsten light balanced, grainy.
K_SFX200T: Special effects film, medium grain.

printFilm: *Popup menu, Default: Kodak_2383.*
Selects the print film stock.

None: Ignore any effect of the print film. This causes the negative to be output directly. If the negative film is also set to None, the color correction and grain are disabled.
Kodak_2383: Kodak VISION Color Print Film, rich blacks.
Kodak_2393: Kodak VISION Premier Color Print Film, rich blacks, some grain.
Kodak_2395: Kodak VISION Color Teleprint Film, low contrast.
Kodak_5386: Eastman EXR Color Print Film (discontinued by Kodak, replaced by 2383).

defield: *Popup menu, Default: No.*
Allows removing field artifacts from the input clip. This is useful if you want the clip to look like it was shot on frames instead of fields. You can show a single field, merge the two together, or simulate a 3:2 pulldown stutter pattern.

No: leaves the fields unchanged.
Keep Field1: shows field1 only, removes field2.
Keep Field2: shows field2 only, removes field1.
Merge: blends both fields together to remove interlacing artifacts.

blurInput: *Default:* 0, *Range:* 0 or greater.
The input is smoothed by this amount. This can be used to remove video noise or compression artifacts before processing.

scaleCC: *Default:* 1, *Range:* 0 to 5.
Scales the amount of color correction performed due to the film types, gamma values, and exposure values. Set to 0 to disable color correction. If you increase this above 1.0 it exaggerates the color correction, which normally increases the contrast.

inputGamma: *Default:* 2.2, *Range:* 0.1 or greater.
The gamma that your original clip was shot for. For video this is normally 2.2; for synthetic computer graphics it may be less.

outputGamma: *Default:* 2.2, *Range:* 0.1 or greater.
The intended viewing gamma of the output.

negExposure: *Default:* 0, *Range:* any.
Adjusts the simulated exposure of the negative film, in stops. Increase for over−exposed and brighter.

printExposure: *Default:* 0, *Range:* any.
Adjusts the simulated exposure of the print film, in stops. Increase for over−exposed and darker.

printLightsRed: *Default:* 25, *Range:* 0 to 50.
Adjusts the red exposure of the print film, in printer light points. 1 light point is 1/12 stop. Increase to over−expose red and give a more cyan result.

printLightsGreen: *Default:* 25, *Range:* 0 to 50.
Adjusts the green exposure of the print film, in printer light points. 1 light point is 1/12 stop. Increase to over−expose green and give a more magenta result.

printLightsBlue: *Default:* 25, *Range:* 0 to 50.
Adjusts the blue exposure of the print film, in printer light points. 1 light point is 1/12 stop. Increase to over−expose blue and give a more yellow result.

offsetDarks: *Default:* 0, *Range:* any.
Adds this gray value to the darker regions of the final result after the other color correction, glow, and grain are applied. This can be negative to increase contrast. (This parameter is not affected by Scale CC.)

scaleBrights: *Default:* 1, *Range:* 0 or greater.
Scales the bright areas of the final result after the other color correction, glow, and grain are applied. (This parameter is not affected by Scale CC.)

glowBrightness: *Default:* 0, *Range:* 0 or greater.
If positive, the image is combined with a blurred version of itself to give a glowing look. Increase for a brighter glow.

glowSoftFocus: *Default:* 0, *Range:* 0 to 1.
If positive, the image is mixed with a blurred version of itself to give a soft focus look. The effect of this parameter is similar to Glow Brightness, but this does not brighten the overall result. Increase this to mix in more of the blurred version and less of the original. If this is 1 and Glow Brightness is 0 you will get only the blurred version.

glowWidth: *Default:* .025*width, *Range:* 0 or greater.
The width of the blur used by the glow and/or soft focus.

glowWidthRed: *Default:* 1, *Range:* 0 or greater.
The relative glow width for the red channel.

glowWidthGreen: *Default:* 1, *Range:* 0 or greater.
The relative glow width for the green channel.

glowWidthBlue: *Default:* 1, *Range:* 0 or greater.
The relative glow width for the blue channel.

grainAmp: *Default:* 1, *Range:* 0 or greater.
Scales the amplitude of the film grain that is added to the result. Set this to 0 to disable all grain.

grainAmpRed: *Default:* 0.9, *Range:* 0 or greater.
Scales the red grain amplitude.

grainAmpGreen: *Default:* 1, *Range:* 0 or greater.
Scales the green grain amplitude.

grainAmpBlue: *Default:* 1.6, *Range:* 0 or greater.
Scales the blue grain amplitude. Note that grain is added and subtracted from the image, so for example, increasing Grain Amp Blue will amplify both the blue and yellow speckles.

grainAmpDarks: *Default:* 0.2, *Range:* 0 to 2.
The relative amount of grain applied to the darkest regions of the image, per channel. This defaults to less than 1.0 because dark areas usually have less grain than midtones.

grainAmpBrights: *Default:* 0, *Range:* 0 to 2.
The relative amount of grain applied to the brightest regions of the image, per channel. This defaults to zero because bright areas usually have less grain than midtones. Note that highly saturated colors can be affected by both Grain Amp Darks and Grain Amp Brights because they are dark in some color channels and bright in others.

midtonePosRed: *Default:* 0.5, *Range:* 0 to 1.
The position of the midtones in the red channel that will normally receive the maximum amount of grain. The red grain amplitude is interpolated from Grain Amp Darks at black, up to 1.0 at this midtone position, then down to Grain Amp Brights at white. This whole curve is then scaled by the Grain Amp Red parameter.

midtonePosGreen: *Default:* 0.5, *Range:* 0 to 1.
The position of the midtones in the green channel that will normally receive the maximum amount of grain. The green grain amplitude is interpolated from Grain Amp Darks at black, up to 1.0 at this midtone position, then down to Grain Amp Brights at white. This whole curve is then scaled by the Grain Amp Green parameter.

midtonePosBlue: *Default:* 0.5, *Range:* 0 to 1.
The position of the midtones in the blue channel that will normally receive the maximum amount of grain. The blue grain amplitude is interpolated from Grain Amp Darks at black, up to 1.0 at this midtone position, then down to Grain Amp Brights at white. This whole curve is then scaled by the Grain Amp Blue parameter.

grainBlur: *Default:* 0, *Range:* 0 or greater.
The grain is smoothed by this amount. Increase for coarser grain.

grainBlurRed: *Default:* 1, *Range:* 0 or greater.
The relative blur amount for the red grain.

grainBlurGreen: *Default:* 0.9, *Range:* 0 or greater.
The relative blur amount for the green grain.

grainBlurBlue: *Default:* 1.2, *Range:* 0 or greater.
The relative blur amount for the blue grain.

grainMono: *Toggle−button, Default:* off.
When enabled, the same grain pattern is used for the red, green, and blue channels. To make truly monochrome grain you should also set Grain Amp Red/Green/Blue equal to each other, make sure Midtone Pos Red/Green/Blue are equal, and if GrainBlur is positive also set Grain Blur Red/Green/Blue equal

grainHold: *Popup menu, Default: Frame.*
Indicates how often a new grain pattern should be generated. You will probably only notice a difference between these options if Grain Blur is positive to make the grain size larger than one pixel.

Field: holds the grain pattern for one field.

Frame: holds the grain pattern for one frame (2 fields).
3:2 Pulldown at 0: holds the grain in a 3:2 pulldown pattern with the first pulldown frame at 0. These options are appropriate if your clip was created at 24 fps but is now in 30 fps pulldown form. They will not make sense if your clip is 24P. A 3:2 pulldown pattern repeats every 5 frames, so if frame 1:00:23 is the first frame with field artifacts after three normal frames, then you should specify 3 as the first pulldown frame.
3:2 Pulldown at 1: holds the grain in a 3:2 pulldown pattern with the first pulldown frame at 1.
3:2 Pulldown at 2: holds the grain in a 3:2 pulldown pattern with the first pulldown frame at 2.
3:2 Pulldown at 3: holds the grain in a 3:2 pulldown pattern with the first pulldown frame at 3.
3:2 Pulldown at 4: holds the grain in a 3:2 pulldown pattern with the first pulldown frame at 4.

grainSeed: *Default:* 0.123, *Range:* 0 or greater.
Initializes the random number generator for the grain generation. The actual seed value is not significant, but different seeds give different grain patterns and the same value should give a repeatable pattern.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_FilmEffect(Source, 1.06, negFilm, etc...);`

See Also:

Grain
Static
Diffuse
Clouds
DissolveSpeckle
Sapphire Plug−ins Introduction

Flicker

Scales the colors of the source clip by different amounts over time for a flickering effect. The pattern of flickering can be random, a periodic wave, or a combination of the two.

Inputs:

Source: The clip to be processed.

Parameters:

randAmp: *Default:* 0.2, *Range:* 0 or greater.
The amplitude of smooth but random flickering.

randFreq: *Default:* 30, *Range:* 0 or greater.
The frequency of the random flickering. Increase for more variation between frames. Decrease for slower flickering.

waveAmp: *Default:* 0, *Range:* 0 or greater.
The amplitude of periodic wave flickering.

waveFreq: *Default:* 5, *Range:* 0 or greater.
The frequency of the wave flickering. Increase for faster flickering, decrease for slower. This has no effect if Wave Amp is 0.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

seed: *Default:* 0.123, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

Script Form: `S_Flicker(Source, 1.06, randAmp, etc...);`

See Also:

FlickerRemove
FlickerMatch
Shake
HueSatBright
Sapphire Plug−ins Introduction

FlickerMatch

In the S_FlickerMatch Plugin.

Adds flicker to the Source clip using the flicker from a second Match clip. For example, a clip can be brightened in synchrony with a flashing light in another clip. To use this effect, first position the corners of the rectangle over an area of the Match clip which has brightness changes you want to copy. A middle or light gray area is best for this. Then select a frame where you want the Source brightness unchanged, and hit the Set Match Level button. When other frames are processed, the Source brightness will be scaled by the average Match brightness within the rectangle, relative to the Match Level.

Inputs:

Source: The clip to add flicker to.

Match: The clip to copy flicker from.

Parameters:

rectCorner1: *X & Y*, *Default:* [.2*width .2*height], *Range:* any.
The upper left corner of the rectangle which is used to measure the flicker, in screen coordinates.

rectCorner2: *X & Y*, *Default:* [.8*width .8*height], *Range:* any.
The lower right corner of the rectangle which is used to measure the flicker, in screen coordinates.

matchLevel: *Default:* 0.5, *Range:* 0.01 or greater.
The average Match brightness in the rectangle for which the Source input is unchanged.

setMatchLevel: *Push−button*.
Pressing this button sets the Match Level parameter to the average Match clip brightness in the rectangle at the current frame. It causes the output to equal the Source at this frame.

Script Form: `S_FlickerMatch(Source, Match, 1.06, "Flicker Match", rectCorner1X, etc...);`

See Also:

FlickerMatchColor

FlickerRemove
FlickerRemoveColor
FlickerRemoveMatte
FlickerRmMatteColor
FlickerMatchMatte
FlickerMchMatteColor
Flicker
Sapphire Plug−ins Introduction

FlickerMatchMatte

In the S_FlickerMchMatte Plugin.

Adds flicker to the Source clip using the flicker from a second Match clip, in the areas specified by a Matte. To use this effect, select a frame where you want the Source brightness unchanged, and hit the Set Match Level button. When other frames are processed, the Source brightness will be scaled by the average Match brightness within the Matte, relative to the Match Level.

Inputs:

Source: The clip to add flicker to.

Match: The clip to copy flicker from.

Matte: *Optional.* This clip specifies which Source areas to measure the flicker from. If this input is not provided, the Alpha of the Match input is used as the Matte instead. It can be inverted with the Invert Matte parameter.

Parameters:

matchLevel: *Default:* 0.5, *Range:* 0.01 or greater.
The average Match brightness in the Matte for which the Source input is unchanged.

setMatchLevel: *Push−button.*
Pressing this button sets the Match Level parameter to the average Match clip brightness within the Matte at the current frame. It causes the output to equal the Source at this frame.

invertMatte: *Toggle−button, Default:* off.
If enabled, the black and white of the matte are inverted before use.

Script Form: `S_FlickerMchMatte(Source, Match, Matte, 1.06, "Flicker Match Matte", matchLevel, etc...);`

See Also:

FlickerMchMatteColor

FlickerRemove
FlickerRemoveColor
FlickerRemoveMatte
FlickerRmMatteColor
FlickerMatch
FlickerMatchColor
Flicker
Sapphire Plug−ins Introduction

FlickerMatchColor

In the S_FlickerMatch Plugin.

Adds color changes to the Source clip using the color changes from a second Match clip. Similar to FlickerMatch but the process is applied to each color channel. To use this effect, first position the corners of the rectangle over an area of the Match clip which has color changes you want to copy. A middle or light gray area is best for this. Then select a frame for which you want the Source color unchanged, and hit the Set Match Level button. When you process other frames, the Source colors will be scaled by the average Match color within the rectangle, relative to the Match Color.

Inputs:

Source: The clip to add color changes to.

Match: The clip to copy color changes from.

Parameters:

rectCorner1: *X & Y, Default:* [.2*width .2*height], *Range:* any.
The upper left corner of the rectangle which is used to measure the flicker, in screen coordinates.

rectCorner2: *X & Y, Default:* [.8*width .8*height], *Range:* any.
The lower right corner of the rectangle which is used to measure the flicker, in screen coordinates.

matchColor: *Default rgb:* [0.5 0.5 0.5].
The average Match color in the rectangle for which the Source input is unchanged.

setMatchColor: *Push−button.*
Pressing this button sets the Match Color parameter to the average Match clip color in the rectangle at the current frame. It causes the output to equal the Source at this frame.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_FlickerMatch(Source, Match, 1.06, "Flicker Match Color", rectCorner1X, etc...);`

See Also:

FlickerMatch

FlickerRemove
FlickerRemoveColor
FlickerRemoveMatte
FlickerRmMatteColor
FlickerMatchMatte
FlickerMchMatteColor
Flicker
Sapphire Plug−ins Introduction

FlickerMchMatteColor

In the S_FlickerMchMatte Plugin.

Adds color changes to the Source clip using the color changes from a second Match clip, in the areas specified by a Matte. To use this effect, select a frame where you want the Source color unchanged, and hit the Set Match Color button. When other frames are processed, the Source color will be scaled by the average Match color within the Matte, relative to the Match Color.

Inputs:

Source: The clip to add flicker to.

Match: The clip to copy flicker from.

Matte: *Optional.* This clip specifies which Source areas to measure the flicker from. If this input is not provided, the Alpha of the Match input is used as the Matte instead. It can be inverted with the Invert Matte parameter.

Parameters:

matchColor: *Default rgb:* [0.5 0.5 0.5].
The average Match color in the Matte for which the Source input is unchanged.

setMatchColor: *Push−button.*
Pressing this button sets the Match Color parameter to the average Match clip color within the Matte at the current frame. It causes the output to equal the Source at this frame.

invertMatte: *Toggle−button, Default:* off.
If enabled, the black and white of the matte are inverted before use.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_FlickerMchMatte(Source, Match, Matte, 1.06, "Flicker Mch Matte Color", matchColorRed, etc...);`

See Also:

FlickerMatchMatte

FlickerRemove
FlickerRemoveColor
FlickerRemoveMatte
FlickerRmMatteColor
FlickerMatch
FlickerMatchColor
Flicker
Sapphire Plug−ins Introduction

FlickerRemove

In the S_FlickerRemove Plugin.

Removes temporal flickering from the Source clip. For example, old footage with uneven exposure times can be smoothed out with this effect. To use this effect, first position the corners of the rectangle over an area where the average brightness should remain constant. A middle or light gray area is best for this. Then select a Source frame that has the desired brightness within the rectangle, and hit the Set Hold Level button. When other frames are processed, their brightness will be scaled so the average brightness within the rectangle is equal to the Hold Level. You can keyframe different Hold Level values over time to account for desirable brightness changes.

Inputs:

Source: The clip to remove flicker from.

Parameters:

rectCorner1: *X & Y, Default:* [.2*width .2*height], *Range:* any.
The upper left corner of the rectangle which is used to measure the flicker, in screen coordinates.

rectCorner2: *X & Y, Default:* [.8*width .8*height], *Range:* any.
The lower right corner of the rectangle which is used to measure the flicker, in screen coordinates.

holdLevel: *Default:* 0.5, *Range:* 0.01 or greater.
The requested average output brightness for the area within the rectangle.

setHoldLevel: *Push−button.*
Pressing this button has a side effect of setting the Hold Level parameter to the average Source brightness in the rectangle at the current frame. It causes the output to equal the Source at this frame. This button retains no value itself, and is turned back off immediately after being pushed.

Script Form: `S_FlickerRemove(Source, 1.06, "Flicker Remove", rectCorner1X, etc...);`

See Also:

FlickerRemoveColor
FlickerRemoveMatte
FlickerRmMatteColor
FlickerMatch
FlickerMatchColor
FlickerMatchMatte
FlickerMchMatteColor
Flicker
Sapphire Plug−ins Introduction

FlickerRemoveMatte

In the S_FlickerRmMatte Plugin.

Removes temporal flickering from the Source clip using a Matte clip to specify the area where the average brightness should remain constant. To use this effect, select a Source frame that has the desired brightness within the Matte, and hit the Set Hold Level button. When other frames are processed, their brightness will be scaled so the average brightness within the Matte is equal to the Hold Level. You can keyframe different Hold Level values over time to account for desirable brightness changes.

Inputs:

Source: The clip to remove flicker from.

Matte: *Optional.* This clip specifies which Source areas to measure the flicker from. If this input is not provided, the Alpha of the Source input is used as the Matte instead. It can be inverted with the Invert Matte parameter.

Parameters:

holdLevel: *Default:* 0.5, *Range:* 0.01 or greater.
The requested average output brightness for the area within the Matte.

setHoldLevel: *Push−button.*
Pressing this button has a side effect of setting the Hold Level parameter to the average Source brightness within the Matte at the current frame. It causes the output to equal the Source at this frame. This button retains no value itself, and is turned back off immediately after being pushed.

invertMatte: *Toggle−button, Default:* off.
If enabled, the black and white of the matte are inverted before use.

Script Form: `S_FlickerRmMatte(Source, Matte, 1.06, "Flicker Remove Matte", holdLevel, etc...);`

See Also:

FlickerRmMatteColor

FlickerRemove
FlickerRemoveColor
FlickerMatch
FlickerMatchColor
FlickerMatchMatte
FlickerMchMatteColor
Flicker
Sapphire Plug−ins Introduction

FlickerRemoveColor

In the S_FlickerRemove Plugin.

Removes temporal color changes from the Source clip. Similar to FlickerRemove but the process is applied to each color channel. To use this effect, first position the corners of the rectangle over an area where the average color should remain constant. A middle or light gray area is best for this. Then select a Source frame that has the desired color within the rectangle, and hit the Set Hold Color button. When other frames are processed, their colors will be scaled so the average color within the rectangle is equal to the Hold Color.

Inputs:

Source: The clip to remove color changes from.

Parameters:

rectCorner1: *X & Y*, *Default:* [.2*width .2*height], *Range:* any.
The upper left corner of the rectangle which is used to measure the flicker, in screen coordinates.

rectCorner2: *X & Y*, *Default:* [.8*width .8*height], *Range:* any.
The lower right corner of the rectangle which is used to measure the flicker, in screen coordinates.

holdColor: *Default rgb:* [0.5 0.5 0.5].
The requested average output color for the area within the rectangle.

setHoldColor: *Push−button*.
Pressing this button has a side effect of setting the Hold Color parameter to the average Source color in the rectangle at the current frame. It causes the output to equal the Source at this frame. This button retains no value itself, and is turned back off immediately after being pushed.

Script Form: `S_FlickerRemove(Source, 1.06, "Flicker Remove Color", rectCorner1X, etc...);`

See Also:

FlickerRemove
FlickerRemoveMatte
FlickerRmMatteColor
FlickerMatch
FlickerMatchColor
FlickerMatchMatte
FlickerMchMatteColor
Flicker
Sapphire Plug−ins Introduction

FlickerRmMatteColor

In the S_FlickerRmMatte Plugin.

Removes temporal color changes from the Source clip using a Matte clip to specify the area where the average color should remain constant. To use this effect, select a Source frame that has the desired color within the Matte, and hit the Set Hold Color button. When other frames are processed, their color will be scaled so the average color within the Matte is equal to the Hold Color.

Inputs:

Source: The clip to remove color changes from.

Matte: *Optional.* This clip specifies which Source areas to measure the flicker from. If this input is not provided, the Alpha of the Source input is used as the Matte instead. It can be inverted with the Invert Matte parameter.

Parameters:

holdColor: *Default rgb:* [0.5 0.5 0.5].
The requested average output color for the area within the Matte.

setHoldColor: *Push−button.*
Pressing this button has a side effect of setting the Hold Color parameter to the average Source color within the Matte at the current frame. It causes the output to equal the Source at this frame. This button retains no value itself, and is turned back off immediately after being pushed.

invertMatte: *Toggle−button, Default:* off.
If enabled, the black and white of the matte are inverted before use.

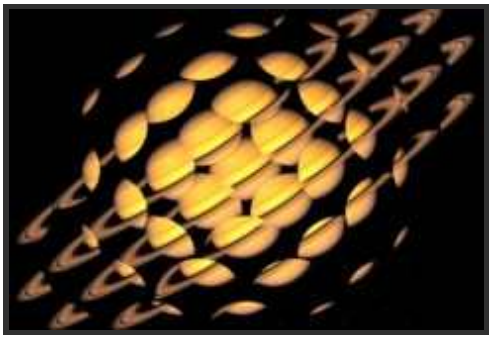
Script Form: `S_FlickerRmMatte(Source, Matte, 1.06, "Flicker Rm Matte Color", holdColorRed, etc...);`

See Also:

FlickerRemoveMatte

FlickerRemove
FlickerRemoveColor
FlickerMatch
FlickerMatchColor
FlickerMatchMatte
FlickerMchMatteColor
Flicker
Sapphire Plug−ins Introduction

FlysEyeCircles



In the S_FlysEye Plugin.

Breaks the image into circle shaped tiles and transforms the image within each shape, to create a fly's eye view effect. The Overlap options allow the circles to be combined in different ways where they overlap. The 'Inside' parameters transform the Source image before it is tiled into the pattern, and the 'Tile' parameters transform the entire fly's eye pattern.

Inputs:

Source: The clip to be processed.

Parameters:

tileFrequency: *Default:* 12, *Range:* 0.1 or greater.
The frequency of the tile pattern, increase for more smaller tiles.

tileRelHeight: *Default:* 1, *Range:* 0.01 or greater.
The relative height of the tile shapes, increase for taller tiles.

tileShift: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translates the tile pattern.

circleRadius: *Default:* 1, *Range:* 0 to 1.
The radius of the circles relative to each other. If this is less than 1.0 you will get empty spaces between the circles. The color of these empty spaces will be either black or white depending on the combine mode.

edgeSoftness: *Default:* 0, *Range:* 0 to 1.
The softness of the edges of the circles. If this is increased, it may also be necessary to lower the Circle Radius to avoid rectangular artifacts where the soft edges overlap.

insideZdist: *Default:* 2, *Range:* 0 or greater.
Determines the zoom factor of the image inside each tile. Values greater than 1 zoom out, values less than 1 zoom in. If this is 1, Inside Rotate is 0, and Overall Zdist is 1, the result should be the same as the input image.

insideRotate: *Default:* 0, *Range:* any.
The rotation angle of the image inside each tile, in degrees.

overallZdist: *Default:* 1, *Range:* any.
Creates an overall zooming effect by making each tile look toward or away from the image center. Decrease to zoom in, increase to zoom out. When 0 all tiles should contain identical images.

circleOverlap: *Popup menu, Default: Ave.*
Determines the method used to combine the overlapping regions of the circles.

Ave: uses a weighted average across the overlapping region for a smooth transition.
Screen: uses a screen operation.
Max: uses the lighter.
Min: uses the darker.
Mult: uses a multiply operation.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

wrap: *Radio buttons, Default: Reflect.*
Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

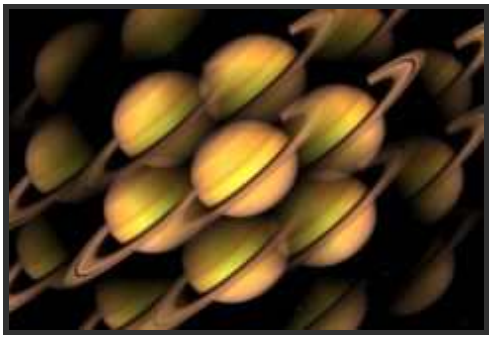
filter: *Toggle−button, Default:* on.
If enabled, the Source image is resampled using pixel averaging. This removes aliasing and gives a higher quality result especially when Inside Zdist is large. It may not be necessary if your input image is smooth or Inside Zdist is small.

Script Form: `S_FlysEye(Source, 1.06, 2, tileFrequency, etc...);`

See Also:

FlysEyeHex
FlysEyeRect
Mosaic
HalfTone
ScanLines
JpegDamage
VanGogh
Sapphire Plug−ins Introduction

FlysEyeHex



In the S_FlysEye Plugin.

Breaks the image into hexagon shaped tiles and transforms the image within each shape, to create a fly's eye view effect. Increase Edge Softness for a smoother overlap between the tiles. The 'Inside' parameters transform the Source image before it is tiled into the pattern, and the 'Tile' parameters transform the entire fly's eye pattern.

Inputs:

Source: The clip to be processed.

Parameters:

tileFrequency: *Default:* 12, *Range:* 0.1 or greater.
The frequency of the tile pattern, increase for more smaller tiles.

tileRelHeight: *Default:* 1, *Range:* 0.01 or greater.
The relative height of the tile shapes, increase for taller tiles.

tileShift: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translates the tile pattern.

tileRotate: *Default:* 0, *Range:* any.
The rotation angle of the tile pattern, in degrees.

edgeSoftness: *Default:* 0, *Range:* 0 to 1.
The softness of the edges between the tile shapes. Increase for smoother blending between the shapes.

insideZdist: *Default:* 2, *Range:* 0 or greater.
Determines the zoom factor of the image inside each tile. Values greater than 1 zoom out, values less than 1 zoom in. If this is 1, Inside Rotate is 0, and Overall Zdist is 1, the result should be the same as the input image.

insideRotate: *Default:* 0, *Range:* any.
The rotation angle of the image inside each tile, in degrees.

overallZdist: *Default:* 1, *Range:* any.
Creates an overall zooming effect by making each tile look toward or away from the image center. Decrease to zoom in, increase to zoom out. When 0 all tiles should contain identical images.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

wrap: *Radio buttons, Default: Reflect.*
Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

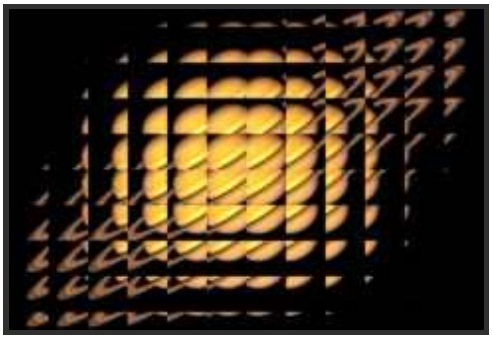
filter: *Toggle−button, Default:* on.
If enabled, the Source image is resampled using pixel averaging. This removes aliasing and gives a higher quality result especially when Inside Zdist is large. It may not be necessary if your input image is smooth or Inside Zdist is small.

Script Form: `S_FlysEye(Source, 1.06, 0, tileFrequency, etc...);`

See Also:

FlysEyeRect
FlysEyeCircles
Mosaic
HalfTone
ScanLines
JpegDamage
VanGogh
Sapphire Plug−ins Introduction

FlysEyeRect



In the S_FlysEye Plugin.

Breaks the image into rectangle shaped tiles and transforms the image within each shape, to create a fly's eye view effect. The 'Inside' parameters transform the Source image before it is tiled into the pattern, and the 'Tile' parameters transform the entire fly's eye pattern.

Inputs:

Source: The clip to be processed.

Parameters:

tileFrequency: *Default:* 12, *Range:* 0.1 or greater.
The frequency of the tile pattern, increase for more smaller tiles.

tileRelHeight: *Default:* 1, *Range:* 0.01 or greater.
The relative height of the tile shapes, increase for taller tiles.

tileShift: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translates the tile pattern.

insideZdist: *Default:* 2, *Range:* 0 or greater.
Determines the zoom factor of the image inside each tile. Values greater than 1 zoom out, values less than 1 zoom in. If this is 1, Inside Rotate is 0, and Overall Zdist is 1, the result should be the same as the input image.

insideRotate: *Default:* 0, *Range:* any.
The rotation angle of the image inside each tile, in degrees.

overallZdist: *Default:* 1, *Range:* any.
Creates an overall zooming effect by making each tile look toward or away from the image center. Decrease to zoom in, increase to zoom out. When 0 all tiles should contain identical images.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

wrap: *Radio buttons, Default: Reflect.*
Determines the method for accessing outside the borders of the source image.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the Source image is resampled using pixel averaging. This removes aliasing and gives a higher quality result especially when Inside Zdist is large. It may not be necessary if your input image is smooth or Inside Zdist is small.

Script Form: `S_FlysEye(Source, 1.06, 1, tileFrequency, etc...);`

See Also:

FlysEyeHex
FlysEyeCircles
Mosaic
HalfTone
ScanLines
JpegDamage
VanGogh
Sapphire Plug−ins Introduction

Glare

Composites rainbow halos and/or glint−like rays over the Background clip at locations where the Source clip is brighter than the threshold. Glares are best observed on dark images with a few bright spots.

Inputs:

Source: The input clip that determines the glare locations and colors.

Background: *Optional.* The clip to combine the glares with. If no background is given, the Source is also used as the Background.

S_Mask: *Optional.* If connected, the source glare colors are scaled by this input. A monochrome Mask can be used to choose a subset of Source areas that will generate glares. A color Mask can be used to selectively adjust the glare colors in different regions. The Mask is applied to the source before the glares are generated so it will not clip the resulting glares.

Parameters:

type: *Popup menu, Default: rainbow_rays.*
The style of glare to apply. Custom glare types can also be made, or existing types modified, by editing the "s_glares.text" file.

rainbow_rays: rays with rainbow.
rainbow_rays2: rays with larger rainbow.
rays_60: cluster of 60 rays.
rays_20: cluster of 20 rays.
rays_16: cluster of 16 rays.
rays_12_rand: 12 rays with random orientations.
rays_6: cluster of 6 rays.
rays_4: cluster of 4 rays.
rays_4_ring: 4 rays with ring and glow.
rays_multi: several clusters of rays together.
rainbow_only: rainbow.
rainbow2_only: larger rainbow.
rainbow_double: 2 rainbows.
round_coin: simple solid circle.

convolve: *Radio buttons, Default: NO.*
Determines the method for applying the glares to the Background.

NO: by default, the center of each bright spot is found and a glare is rendered at that location. In this mode, if there are multiple moving bright spots that merge or separate, pops in the glare locations can occur.

YES: the convolve option smoothly adds glares for each pixel of the source image above the threshold value. This gives a different look which can be preferable. However, if there are many bright areas needing glares, it can be unbearably slow, so in this mode the processing times are estimated and very slow frames are aborted.

Even if slow: use this option if you are sure you want to convolve all the frames of a clip even if they might be slow to process.

glareRes: *Radio buttons, Default: Full.*
Selects the resolution factor for the glares. Higher resolutions give sharper glares, lower resolutions give smoother glares and faster processing. This 'Res' factor only affects the glares: the background is still combined with the glares at full resolution.

Full: Full resolution is used.
1/2: The glares are calculated at half resolution.
1/4: The glares are calculated at quarter resolution.

size: *Default:* .1*width, *Range:* 0 or greater.
Scales the size of the glares.

relHeight: *Default:* 1, *Range:* 0 or greater.
Scales the vertical dimension of the glares, making them elliptical instead of circular.

rotate: *Default:* 0, *Range:* any.
Rotates the ray elements of the glares, if any, in degrees.

scaleColors: *Default rgb:* [1 1 1].
Scales the color of the glares. The colors and brightnesses of the glares are also affected by the Source and Mask inputs.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of all the glares.

saturation: *Default:* 1, *Range:* any.
Scales the color saturation of the glare elements. Increase for more intense colors. Set to 0 for monochrome glares.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the glares. The maximum of the red, green, and blue glare brightness is scaled by this value and combined with the background Alpha at each pixel.

thresholdAddColor: *Default rgb:* [0 0 0].
This can be used to raise the threshold on a specific color and thereby reduce the glares generated on areas of the source clip containing that color.

threshold: *Default:* 0.8, *Range:* 0 to 1.
Glares are generated from locations in the source clip that are brighter than this value. A value of 0.9 causes glares at only the brightest spots. A value of 0 causes glares for every non−black area.

thresholdBlur: *Default:* 2, *Range:* 0 or greater.
Increase to smooth out the areas creating glares. This can be used to eliminate glares generated from small speckles or to simply soften the glares. Increasing this may put more highlights below the threshold and darken the resulting glares, but you can decrease the Threshold parameter to compensate.

glareFromAlpha: *Default:* 0, *Range:* 0 to 1.
Set to 1 to generate glares from the alpha channel of the source input instead of the RGB channels. In this case the glares will not pick up color from the source and will typically be brighter. Values between 0 and 1 interpolate between using the RGB and the Alpha.

glareUnderSource: *Default:* 0, *Range:* 0 to 1.
Set to 1 to composite the Source input over the glares.

scaleSource: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the Source input when combined with the glares. This does not affect the generation of the glares themselves.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background. This parameter only has an effect if the background input is provided, and is visible due to a partially transparent Source image or a reduced Scale Source parameter value.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Glare(Source, Background, S_Mask, 1.06, type, etc...);`

See Also:

Glint
Glow
LensFlare
Sapphire Plug−ins Introduction

Glint

In the S_Glint Plugin.

Generates star shaped glints at locations where the Source clip is brighter than the threshold, and combines the glints with the Background clip. Glints are best observed on dark images with a few bright spots.

Inputs:

Source: The input clip that determines the glint locations and colors.

Background: *Optional.* The clip to combine the glints with. If no background is given, the Source is also used as the Background.

S_Mask: *Optional.* If connected, the source glint colors are scaled by this input. A monochrome Mask can be used to choose a subset of Source areas that will generate glints. A color Mask can be used to selectively adjust the glint colors in different regions. The Mask is applied to the source before the glints are generated so it will not clip the resulting glints.

Parameters:

size: *Default:* .25*width, *Range:* 0 or greater.
Scales the length of all glint rays. Note that a zero glint size still enhances the bright areas; set the brightness parameter to zero if you want to pass the Source through unchanged.

sizeX: *Default:* 1, *Range:* 0 or greater.
Scales the length of the horizontal glint rays.

sizeY: *Default:* 1, *Range:* 0 or greater.
Scales the length of the vertical glint rays.

sizeDiag1: *Default:* 0.75, *Range:* 0 or greater.
Scales the length of the diagonal rays from top right to bottom left.

sizeDiag2: *Default:* 0.75, *Range:* 0 or greater.
Scales the length of the diagonal rays from top left to bottom right.

sizeRed: *Default:* 0.5, *Range:* 0 or greater.
Scales the length of the red component of the rays. If the red, green, and blue sizes are equal the glints will be uniform in color and will match the color of the source clip. If they are not equal, the glint colors can vary along the lengths of the rays.

sizeGreen: *Default:* 1, *Range:* 0 or greater.
Scales the length of the green component of the rays.

sizeBlue: *Default:* 1.5, *Range:* 0 or greater.
Scales the length of the blue component of the rays.

scaleColors: *Default rgb:* [1 1 1].
Scales the color of the glints. The colors and brightnesses of the glints are also affected by the Source and Mask inputs.

brightness: *Default:* 1, *Range:* 0 or greater.

Scales the brightness of all the glints.

brightnessX: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the horizontal glint rays.

brightnessY: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the vertical glint rays.

brightnessDiag1: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the diagonal rays from top right to bottom left.

brightnessDiag2: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the diagonal rays from top left to bottom right.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the glints. The maximum of the red, green, and blue glint brightness is scaled by this value and combined with the background Alpha at each pixel.

thresholdAddColor: *Default rgb:* [0 0 0].
This can be used to raise the threshold on a specific color and thereby reduce the glints generated on areas of the source clip containing that color.

threshold: *Default:* 0.7, *Range:* 0 to 1.
Glints are generated from locations in the source clip that are brighter than this value. A value of 0.9 causes glints at only the brightest spots. A value of 0 causes glints for every non−black area.

thresholdBlur: *Default:* 2, *Range:* 0 or greater.
Increase to smooth out the areas creating glints. This can be used to eliminate glints generated from small speckles or to simply soften the glints. Increasing this may put more highlights below the threshold and darken the resulting glints, but you can decrease the Threshold parameter to compensate.

glintFromAlpha: *Default:* 0, *Range:* 0 to 1.
Set to 1 to generate glints from the alpha channel of the source input instead of the RGB channels. In this case the glints will not pick up color from the source and will typically be brighter. Values between 0 and 1 interpolate between using the RGB and the Alpha.

glintUnderSource: *Default:* 0, *Range:* 0 to 1.
Set to 1 to composite the Source input over the glints.

scaleSource: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the Source input when combined with the glints. This does not affect the generation of the glints themselves.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background. This parameter only has an effect if the background input is provided, and is visible due to a partially transparent Source image or a reduced Scale Source parameter value.

pixelAspect2: *Toggle−button, Default:* GetDefaultAspect()−.5.
Enable this to squash the pattern vertically by a factor of 2. This effect is not able to adjust for arbitrary pixel aspect ratios, but you can select between pixel aspects of 1 or 2.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Glint(Source, Background, S_Mask, 1.06, "Glint", size, etc...);`

See Also:

GlintRainbow

Glare
Sparkles
Glow
Sapphire Plug−ins Introduction

GlintRainbow

In the S_Glint Plugin.

Generates star shaped rainbow colored glints at locations where the Source clip is brighter than the threshold, and combines the glints with the Background clip. Glints are best observed on dark images with a few bright spots.

Inputs:

Source: The input clip that determines the glint locations and colors.

Background: *Optional.* The clip to combine the glints with. If no background is given, the Source is also used as the Background.

S_Mask: *Optional.* If connected, the source glint colors are scaled by this input. A monochrome Mask can be used to choose a subset of Source areas that will generate glints. A color Mask can be used to selectively adjust the glint colors in different regions. The Mask is applied to the source before the glints are generated so it will not clip the resulting glints.

Parameters:

size: *Default:* .25*width, *Range:* 0 or greater.
Scales the length of all glint rays. Note that a zero glint size still enhances the bright areas; set the brightness parameter to zero if you want to pass the Source through unchanged.

sizeX: *Default:* 1, *Range:* 0 or greater.
Scales the length of the horizontal glint rays.

sizeY: *Default:* 1, *Range:* 0 or greater.
Scales the length of the vertical glint rays.

sizeDiag1: *Default:* 0.75, *Range:* 0 or greater.
Scales the length of the diagonal rays from top right to bottom left.

sizeDiag2: *Default:* 0.75, *Range:* 0 or greater.
Scales the length of the diagonal rays from top left to bottom right.

shiftOut: *Default:* 1, *Range:* any.
Shifts the glint rays outwards from their source highlights by this amount relative to the glint size.

shiftRed: *Default:* 0.3, *Range:* any.
Shifts the red component of the glints in or out relative to the blue. The green is centered between blue and red for a complete spectrum.

shiftBlue: *Default:* −0.3, *Range:* any.
Shifts the blue component of the glints in or out relative to the red and green. This can be used with Shift Red to adjust the range of hues in the glints.

scaleColors: *Default rgb:* [1 1 1].
Scales the color of the glints. The colors and brightnesses of the glints are also affected by the Source and Mask inputs.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of all the glints.

brightnessX: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the horizontal glint rays.

brightnessY: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the vertical glint rays.

brightnessDiag1: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the diagonal rays from top right to bottom left.

brightnessDiag2: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the diagonal rays from top left to bottom right.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the glints. The maximum of the red, green, and blue glint brightness is scaled by this value and combined with the background Alpha at each pixel.

thresholdAddColor: *Default rgb:* [0 0 0].
This can be used to raise the threshold on a specific color and thereby reduce the glints generated on areas of the source clip containing that color.

threshold: *Default:* 0.7, *Range:* 0 to 1.
Glints are generated from locations in the source clip that are brighter than this value. A value of 0.9 causes glints at only the brightest spots. A value of 0 causes glints for every non−black area.

thresholdBlur: *Default:* 2, *Range:* 0 or greater.
Increase to smooth out the areas creating glints. This can be used to eliminate glints generated from small speckles or to simply soften the glints. Increasing this may put more highlights below the threshold and darken the resulting glints, but you can decrease the Threshold parameter to compensate.

glintFromAlpha: *Default:* 0, *Range:* 0 to 1.
Set to 1 to generate glints from the alpha channel of the source input instead of the RGB channels. In this case the glints will not pick up color from the source and will typically be brighter. Values between 0 and 1 interpolate between using the RGB and the Alpha.

glintUnderSource: *Default:* 0, *Range:* 0 to 1.
Set to 1 to composite the Source input over the glints.

scaleSource: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the Source input when combined with the glints. This does not affect the generation of the glints themselves.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background. This parameter only has an effect if the background input is provided, and is visible due to a partially transparent Source image or a reduced Scale Source parameter value.

pixelAspect2: *Toggle−button, Default:* GetDefaultAspect()−.5.
Enable this to squash the pattern vertically by a factor of 2. This effect is not able to adjust for arbitrary pixel aspect ratios, but you can select between pixel aspects of 1 or 2.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Glint(Source, Background, S_Mask, 1.06, "Glint Rainbow", size, etc...);`

See Also:

Glint

Glare
Sparkles
Glow
Sapphire Plug−ins Introduction

Glow

In the S_Glows Plugin.

Areas of the source clip that are brighter than the given threshold are glowed and then combined with the source clip.

Inputs:

Source: The input clip that determines the glow locations and colors. This clip is also used as the background and the glows are layered back onto it.

Background: *Optional.* The clip to combine the glows with. If no background is given, the Source is also used as the Background.

S_Mask: *Optional.* If connected, the source glow colors are scaled by this input. A monochrome Mask can be used to choose a subset of Source areas that will generate glows. A color Mask can be used to selectively adjust the glow colors in different regions. The Mask is applied to the source before the glows are generated so it will not clip the resulting glows.

Parameters:

glowWidth: *Default:* .05*width, *Range:* 0 or greater.
Scales the glow distance. Note that a zero glow width still enhances the bright areas; set the brightness parameter to zero if you want to pass the Source through unchanged.

widthX: *Default:* 1, *Range:* 0 or greater.
Scales the horizontal glow width. Set to 0 for vertical only.

widthY: *Default:* 1/GetDefaultAspect(), *Range:* 0 or greater.
Scales the vertical glow width. Set to 0 for horizontal only.

widthRed: *Default:* 1, *Range:* 0 or greater.
Scales the red glow width. If the red, green, and blue widths are equal, the glows will match the color of the source clip. If they are not equal, the glows will vary in color with distance.

widthGreen: *Default:* 1.2, *Range:* 0 or greater.
Scales the green glow width.

widthBlue: *Default:* 1.4, *Range:* 0 or greater.
Scales the blue glow width.

subpixel: *Toggle−button, Default:* off.
Enables glowing by subpixel widths. Use this for smoother animation of the Width parameters.

color: *Default rgb:* [1 1 1].
Scales the color of the glows. The colors and brightnesses of the glows are also affected by the Source and Mask inputs.

brightness: *Default:* 2, *Range:* 0 or greater.
Scales the brightness of all the glows.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the glows. The maximum of the

red, green, and blue glow brightness is scaled by this value and combined with the background Alpha at each pixel.

thresholdAddColor: *Default rgb:* [0 0 0].
This can be used to raise the threshold on a specific color and thereby reduce the glows generated on areas of the source clip containing that color.

threshold: *Default:* 0, *Range:* 0 to 1.
Glows are generated from locations in the source clip that are brighter than this value. A value of 0.9 causes glows at only the brightest spots. A value of 0 causes glows for every non−black area.

glowFromAlpha: *Default:* 0, *Range:* 0 to 1.
Set to 1 to generate glows from the alpha channel of the source input instead of the RGB channels. In this case the glows will not pick up color from the source and will typically be brighter. Values between 0 and 1 interpolate between using the RGB and the Alpha.

glowUnderSource: *Default:* 0, *Range:* 0 to 1.
Set to 1 to composite the Source input over the glows.

lightBackground: *Default:* 0, *Range:* 0 to 1.
Increase this to give a look of the glow casting light onto the background image. To see this more clearly you can also lower the Scale Background parameter or raise the Brightness parameter.

scaleSource: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the Source input when combined with the glows. This does not affect the generation of the glows themselves.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background. This parameter only has an effect if the background input is provided, and is visible due to a partially transparent Source image or a reduced Scale Source parameter value.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Glows(Source, Background, S_Mask, 1.06, "Glow", glowWidth, etc...);`

See Also:

GlowDist
GlowRainbow
GlowAura
GlowRings
GlowDarks
GlowOrthicon
GlowEdges
GlowNoise
Glint
Sapphire Plug−ins Introduction

GlowAura

In the S_Glows Plugin.

Generates radial colored aura lines following the gradient of the source clip, and then combines these with the source clip.

Inputs:

Source: The input clip that determines the glow locations and directions.

Background: *Optional.* The clip to combine the glows with. If no background is given, the Source is also used as the Background.

S_Mask: *Optional.* If connected, the source glow colors are scaled by this input. A monochrome Mask can be used to choose a subset of Source areas that will generate glows. A color Mask can be used to selectively adjust the glow colors in different regions. The Mask is applied to the source before the glows are generated so it will not clip the resulting glows.

Parameters:

glowWidth: *Default:* .05*width, *Range:* 0 or greater.
Scales the glow distance. Note that a zero glow width still enhances the bright areas; set the brightness parameter to zero if you want to pass the Source through unchanged.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

frequency: *Default:* 8, *Range:* 0 or greater.
The frequency of the color pattern. Increase for more cycles through the spectrum.

frequencyRed: *Default:* 1, *Range:* 0 or greater.
Scales the red frequency.

frequencyGreen: *Default:* 1, *Range:* 0 or greater.
Scales the green frequency.

frequencyBlue: *Default:* 1, *Range:* 0 or greater.
Scales the blue frequency.

subpixel: *Toggle−button, Default:* off.
Enables glowing by subpixel widths. Use this for smoother animation of the Width parameters.

phase: *Default:* 0, *Range:* any.
Shifts the color pattern.

phaseSpeed: *Default:* 1, *Range:* any.
If non−zero, the color phase is automatically animated at this speed, causing the color pattern to flow over time.

phaseRed: *Default:* 0.2, *Range:* any.
Shifts the red phase.

phaseGreen: *Default:* 0.1, *Range:* any.

Shifts the green phase.

phaseBlue: *Default:* 0, *Range:* any.
Shifts the blue phase.

twist: *Default:* 1, *Range:* any.
Adjusts the spiral direction of the radial lines.

color: *Default rgb:* [1 1 1].
Scales the color of the glows.

brightness: *Default:* 0.8, *Range:* 0 or greater.
Scales the brightness of the glows.

outerBrightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the glows at further distances from the source.

glowSaturation: *Default:* 1, *Range:* any.
Scales the saturation of the glow colors. Increase for more intense colors. Set to 0 for monochrome.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the glows. The maximum of the red, green, and blue glow brightness is scaled by this value and combined with the background Alpha at each pixel.

thresholdAddColor: *Default rgb:* [0 0 0].
This can be used to raise the threshold on a specific color and thereby reduce the glows generated on areas of the source clip containing that color.

threshold: *Default:* 0, *Range:* 0 to 1.
Glows will be generated from locations in the source clip that are brighter than this value. A value of 0.9 causes glows at only the brightest spots. A value of 0 causes glows on every non−black area.

glowFromAlpha: *Default:* 0, *Range:* 0 to 1.
Set to 1 to generate glows from the alpha channel of the source input instead of the RGB channels. In this case the glows will not pick up color from the source and will typically be brighter. Values between 0 and 1 interpolate between using the RGB and the Alpha.

glowUnderSource: *Default:* 0, *Range:* 0 to 1.
Set to 1 to composite the Source input over the glows.

lightBackground: *Default:* 0, *Range:* 0 to 1.
Increase this to give a look of the glow casting light onto the background image. To see this more clearly you can also lower the Scale Background parameter or raise the Brightness parameter.

scaleSource: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the Source input when combined with the glows. This does not affect the generation of the glows themselves.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background. This parameter only has an effect if the background input is provided, and is visible due to a partially transparent Source image or a reduced Scale Source parameter value.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Glows(Source, Background, S_Mask, 1.06, "Glow Aura", glowWidth, etc...);`

See Also:

Glow
GlowDist
GlowRainbow
GlowRings
GlowDarks
GlowOrthicon
GlowEdges
GlowNoise
Glint
PsykoStripes
PseudoColor
Sapphire Plug−ins Introduction

GlowDarks

In the S_Glows Plugin.

Areas of the source clip darker than the given threshold are blurred and combined with the input clip to give a deep smoky look. Adjust the Darkness, Width, and Threshold parameters to give different types of looks.

Inputs:

Source: The input clip that determines the glow locations and colors. This clip is also used as the background and the glows are layered back onto it.

Background: *Optional.* The clip to combine the glows with. If no background is given, the Source is also used as the Background.

S_Mask: *Optional.* If connected, the source glow colors are scaled by this input. A monochrome Mask can be used to choose a subset of Source areas that will generate glows. A color Mask can be used to selectively adjust the glow colors in different regions. The Mask is applied to the source before the glows are generated so it will not clip the resulting glows.

Parameters:

glowWidth: *Default:* .1*width, *Range:* 0 or greater.
Scales the glow distance. Note that a zero glow width still affects the dark areas; set the darkness parameter to zero if you want to pass the Source through unchanged.

widthX: *Default:* 1, *Range:* 0 or greater.
Scales the horizontal glow width. Set to 0 for vertical only.

widthY: *Default:* 1/GetDefaultAspect(), *Range:* 0 or greater.
Scales the vertical glow width. Set to 0 for horizontal only.

subpixel: *Toggle−button, Default:* off.
Enables glowing by subpixel widths. Use this for smoother animation of the Width parameters.

darkness: *Default:* 0.5, *Range:* 0 or greater.
The magnitude of the dark glows.

threshold: *Default:* 0.5, *Range:* 0 to 1.
Dark glows will be generated from locations in the source clip that are darker than this value. A value of 0.1 causes glows at only the darkest areas. A value of 1.0 causes glows on every non−white area.

glowSaturation: *Default:* 1, *Range:* any.
Scales the saturation of the dark colors. Increase for more intense colors.

glowFromAlpha: *Default:* 0, *Range:* 0 to 1.
Set to 1 to generate glows from the alpha channel of the source input instead of the RGB channels. In this case the glows will not pick up color from the source and will typically be brighter. Values between 0 and 1 interpolate between using the RGB and the Alpha.

glowUnderSource: *Default:* 0, *Range:* 0 to 1.
Set to 1 to composite the Source input over the glows.

scaleSource: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the Source input when combined with the glows. This does not affect the generation of the glows themselves.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background input clip.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Glows(Source, Background, S_Mask, 1.06, "Glow Darks", glowWidth, etc...);`

See Also:

Glow
GlowDist
GlowRainbow
GlowAura
GlowRings
GlowOrthicon
GlowEdges
GlowNoise
Glint
Sapphire Plug−ins Introduction

GlowDist

In the S_Glows Plugin.

Generates glows based on the distances from the edges of the source input. Any edges in the input image, where the brightness crosses the given threshold value, will generate an equally bright glow into the darker side of the edges. This is best observed when used on images with dark backgrounds.

Inputs:

Source: The input clip that determines the glow locations and colors. This clip is also used as the background and the glows are layered back onto it.

Background: *Optional.* The clip to combine the glows with. If no background is given, the Source is also used as the Background.

S_Mask: *Optional.* If connected, the source glow colors are scaled by this input. A monochrome Mask can be used to choose a subset of Source areas that will generate glows. A color Mask can be used to selectively adjust the glow colors in different regions. The Mask is applied to the source before the glows are generated so it will not clip the resulting glows.

Parameters:

glowWidth: *Default:* .05*width, *Range:* 0 or greater.
Scales the glow distance. Note that a zero glow width still enhances the bright areas; set the brightness parameter to zero if you want to pass the Source through unchanged.

pixelAspect2: *Toggle−button, Default:* GetDefaultAspect()−.5.
Enable this to squash the pattern vertically by a factor of 2. This effect is not able to adjust for arbitrary pixel aspect ratios, but you can select between pixel aspects of 1 or 2.

widthRed: *Default:* 1, *Range:* 0 or greater.
Scales the red glow distance. If the red, green, and blue widths are equal, the glows will be a single color given by the Color parameter. If they are not equal, the glows will vary in color with distance.

widthGreen: *Default:* 1.2, *Range:* 0 or greater.
Scales the green glow width.

widthBlue: *Default:* 1.4, *Range:* 0 or greater.
Scales the blue glow width.

color: *Default rgb:* [1 1 1].
Scales the color of the glows.

brightness: *Default:* 0.8, *Range:* 0 or greater.
Scales the brightness of the glows.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the glows. The maximum of the red, green, and blue glow brightness is scaled by this value and combined with the background Alpha at each pixel.

thresholdAddColor: *Default rgb:* [0 0 0].

This can be used to raise the threshold on a specific color and thereby reduce the glows generated on areas of the source clip containing that color.

threshold: *Default:* 0.5, *Range:* 0 to 1.
Glows are generated at the edges of areas in the source clip that are brighter than this value. A value of 0.9 causes glows from only the brightest spots. A value of 0 causes glows for every non−black area.

glowFromAlpha: *Default:* 0, *Range:* 0 to 1.
Set to 1 to generate glows from the alpha channel of the source input instead of the RGB channels. In this case the glows will not pick up color from the source and will typically be brighter. Values between 0 and 1 interpolate between using the RGB and the Alpha.

glowUnderSource: *Default:* 0, *Range:* 0 to 1.
Set to 1 to composite the Source input over the glows.

lightBackground: *Default:* 0, *Range:* 0 to 1.
Increase this to give a look of the glow casting light onto the background image. To see this more clearly you can also lower the Scale Background parameter or raise the Brightness parameter.

scaleSource: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the Source input when combined with the glows. This does not affect the generation of the glows themselves.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background. This parameter only has an effect if the background input is provided, and is visible due to a partially transparent Source image or a reduced Scale Source parameter value.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Glows(Source, Background, S_Mask, 1.06, "Glow Dist", glowWidth, etc...);`

See Also:

Glow
GlowRainbow
GlowAura
GlowRings
GlowDarks
GlowOrthicon
GlowEdges
GlowNoise
Glint
Sapphire Plug−ins Introduction

GlowEdges

In the S_Glows Plugin.

Glows are created from the edges of the source clip. This differs from the default Glow in that small or thin objects generate as much glow around their edges as large objects. Also the glow colors are not affected by the colors of the source clip.

Inputs:

Source: Edges are extracted from this input clip to determine the glow locations.

Background: *Optional.* The clip to combine the glows with. If no background is given, the Source is also used as the Background.

S_Mask: *Optional.* If connected, the source glow colors are scaled by this input. A monochrome Mask can be used to choose a subset of Source areas that will generate glows. A color Mask can be used to selectively adjust the glow colors in different regions. The Mask is applied to the source before the glows are generated so it will not clip the resulting glows.

Parameters:

glowWidth: *Default:* .025*width, *Range:* 0 or greater.
Scales the glow distance. Note that a zero glow width still enhances the bright areas; set the brightness parameter to zero if you want to pass the Source through unchanged.

widthX: *Default:* 1, *Range:* 0 or greater.
Scales the horizontal glow width. Set to 0 for vertical only.

widthY: *Default:* 1/GetDefaultAspect(), *Range:* 0 or greater.
Scales the vertical glow width. Set to 0 for horizontal only.

widthRed: *Default:* 1, *Range:* 0 or greater.
Scales the red glow width. If the red, green, and blue widths are equal, the glow colors be uniform with distance. If they are not equal, the glows will vary in color with distance.

widthGreen: *Default:* 1.2, *Range:* 0 or greater.
Scales the green glow width.

widthBlue: *Default:* 1.4, *Range:* 0 or greater.
Scales the blue glow width.

subpixel: *Toggle−button, Default:* off.
Enables glowing by subpixel widths. Use this for smoother animation of the Width parameters.

edgesSmooth: *Default:* 0, *Range:* 0 or greater.
Determines the width of the extracted edges which generate the glows.

edgesBrightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the edges before the glows are applied.

edgesThreshold: *Default:* 0, *Range:* 0 or greater.
Increase to remove glows on the less sharp edges.

show: *Radio buttons, Default: Result.*
Selects the type of output.

> ***Result:*** Normally the glows are combined with the source or background, and output.
> ***Edges:*** The edge image only is output, before any glows are applied. This can be helpful while adjusting the various edge parameters.

color: *Default rgb:* [1 1 1].
Scales the color of the glows.

glowBrightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of all the glows.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the glows. The maximum of the red, green, and blue glow brightness is scaled by this value and combined with the background Alpha at each pixel.

glowFromAlpha: *Default:* 0, *Range:* 0 to 1.
Set to 1 to generate glows from the alpha channel of the source input instead of the RGB channels. In this case the glows will not pick up color from the source and will typically be brighter. Values between 0 and 1 interpolate between using the RGB and the Alpha.

glowUnderSource: *Default:* 0, *Range:* 0 to 1.
Set to 1 to composite the Source input over the glows.

lightBackground: *Default:* 0, *Range:* 0 to 1.
Increase this to give a look of the glow casting light onto the background image. To see this more clearly you can also lower the Scale Background parameter or raise the Brightness parameter.

scaleSource: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the Source input when combined with the glows. This does not affect the generation of the glows themselves.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background. This parameter only has an effect if the background input is provided, and is visible due to a partially transparent Source image or a reduced Scale Source parameter value.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Glows(Source, Background, S_Mask, 1.06, "Glow Edges", glowWidth, etc...);`

See Also:

Glow
GlowDist
GlowRainbow
GlowAura
GlowRings
GlowDarks
Glint
Sapphire Plug−ins Introduction

GlowNoise

In the S_Glows Plugin.

Glowing light is generated from areas of the source clip that are brighter than the given threshold. The glows are also attenuated by a solid noise texture to give them a noisy or grainy effect. If the Jitter Frames parameter is positive, the noise will be regenerated for each frame for a fizzling look. If Jitter Frames is zero, two noise textures are combined and slide over each other at a rate depending on the Spread Speed.

Inputs:

Source: The input clip that determines the glow locations and colors. This clip is also used as the background and the glows are layered back onto it.

Background: *Optional.* The clip to combine the glows with. If no background is given, the Source is also used as the Background.

S_Mask: *Optional.* If connected, the source glow colors are scaled by this input. A monochrome Mask can be used to choose a subset of Source areas that will generate glows. A color Mask can be used to selectively adjust the glow colors in different regions. The Mask is applied to the source before the glows are generated so it will not clip the resulting glows.

Parameters:

glowWidth: *Default:* .05*width, *Range:* 0 or greater.
Scales the glow distance. Note that a zero glow width still enhances the bright areas; set the brightness parameter to zero if you want to pass the Source through unchanged.

widthX: *Default:* 1, *Range:* 0 or greater.
Scales the horizontal glow width. Set to 0 for vertical only.

widthY: *Default:* 1/GetDefaultAspect(), *Range:* 0 or greater.
Scales the vertical glow width. Set to 0 for horizontal only.

widthRed: *Default:* 1, *Range:* 0 or greater.
Scales the red glow width. If the red, green, and blue widths are equal, the glows will match the color of the source clip. If they are not equal, the glows will vary in color with distance.

widthGreen: *Default:* 1.2, *Range:* 0 or greater.
Scales the green glow width.

widthBlue: *Default:* 1.4, *Range:* 0 or greater.
Scales the blue glow width.

subpixel: *Toggle−button, Default:* off.
Enables glowing by subpixel widths. Use this for smoother animation of the Width parameters.

noiseAmplitude: *Default:* 1, *Range:* 0 or greater.
The amplitude of noise to include in the glows.

noiseFrequency: *Default:* 40, *Range:* 0.1 or greater.
The spatial frequency of the noise texture. Increase for finer grain, decrease for coarser grain.

noiseFreqRelY: *Default:* GetDefaultAspect(), *Range:* 0.02 or greater.
The relative vertical frequency of the noise texture. Increase to stretch it horizontally or decrease to stretch it vertically.

noiseOctaves: *Integer*, *Default:* 1, *Range:* 1 to 10.
The number of octaves of noise to include. Each octave is twice the frequency and half the amplitude of the previous.

noiseSeed: *Default:* 0.123, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

noiseShift: *X & Y*, *Default:* [0 0], *Range:* any.
The horizontal and vertical translation of the noise texture. This can only be observed if Jitter Frames is zero.

spreadSpeed: *X & Y*, *Default:* [.1*width 0], *Range:* any.
The rate and direction that two noise textures slide over each other. This has no effect unless Jitter Frames is zero.

jitterFrames: *Integer*, *Default:* 1, *Range:* 0 or greater.
If this is 0, the noise texture will remain the same for every frame processed. If it is 1, a new noise texture is used for each frame. If it is 2, a new noise texture is used for every other frame, and so on.

color: *Default rgb:* [1 1 1].
Scales the color of the glows. The colors and brightnesses of the glows are also affected by the Source and Mask inputs.

brightness: *Default:* 2, *Range:* 0 or greater.
Scales the brightness of all the glows.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the glows. The maximum of the red, green, and blue glow brightness is scaled by this value and combined with the background Alpha at each pixel.

thresholdAddColor: *Default rgb:* [0 0 0].
This can be used to raise the threshold on a specific color and thereby reduce the glows generated on areas of the source clip containing that color.

threshold: *Default:* 0, *Range:* 0 to 1.
Glows are generated from locations in the source clip that are brighter than this value. A value of 0.9 causes glows at only the brightest spots. A value of 0 causes glows for every non−black area.

glowFromAlpha: *Default:* 0, *Range:* 0 to 1.
Set to 1 to generate glows from the alpha channel of the source input instead of the RGB channels. In this case the glows will not pick up color from the source and will typically be brighter. Values between 0 and 1 interpolate between using the RGB and the Alpha.

glowUnderSource: *Default:* 0, *Range:* 0 to 1.
Set to 1 to composite the Source input over the glows.

lightBackground: *Default:* 0, *Range:* 0 to 1.
Increase this to give a look of the glow casting light onto the background image. To see this more clearly you can also lower the Scale Background parameter or raise the Brightness parameter.

scaleSource: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the Source input when combined with the glows. This does not affect the generation of the glows themselves.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background. This parameter only has an effect if the background input is provided, and

is visible due to a partially transparent Source image or a reduced Scale Source parameter value.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Glows(Source, Background, S_Mask, 1.06, "Glow Noise", glowWidth, etc...);`

See Also:

Glow
GlowDist
GlowRainbow
GlowAura
GlowRings
GlowDarks
GlowOrthicon
GlowEdges
Glint
Sapphire Plug−ins Introduction

GlowOrthicon

In the S_Glows Plugin.

The source clip is darkened at areas around parts of the source clip that are brighter than the given threshold, to give an 'orthicon' or 'dark glow' look. Lower the Threshold parameter to produce the orthicon effect in more areas. Adjust the Darkness and Width parameters to give different types of looks.

Inputs:

Source: The input clip that determines the locations to be darkened.

Background: *Optional.* The clip to combine the glows with. If no background is given, the Source is also used as the Background.

S_Mask: *Optional.* If connected, the source glow colors are scaled by this input. A monochrome Mask can be used to choose a subset of Source areas that will generate glows. A color Mask can be used to selectively adjust the glow colors in different regions. The Mask is applied to the source before the glows are generated so it will not clip the resulting glows.

Parameters:

darksWidth: *Default:* .025*width, *Range:* 0 or greater.
Scales the dark glow distance.

protectWidth: *Default:* 0.1, *Range:* 0 or greater.
The distance around the bright areas that is protected from darkening. This should normally be less than the value of Darks Width.

protectAmount: *Default:* 1, *Range:* 0 or greater.
The amount that the bright areas are protected from darkening.

widthX: *Default:* 1, *Range:* 0 or greater.
Scales the horizontal glow width. Set to 0 for vertical only.

widthY: *Default:* 1/GetDefaultAspect(), *Range:* 0 or greater.
Scales the vertical glow width. Set to 0 for horizontal only.

subpixel: *Toggle−button, Default:* off.
Enables glowing by subpixel widths. Use this for smoother animation of the Width parameters.

color: *Default rgb:* [0 0 0].
Scales the color of the glows. The colors and brightnesses of the glows are also affected by the Source and Mask inputs.

darkness: *Default:* 1, *Range:* 0 or greater.
Scales the amount of darkening.

thresholdAddColor: *Default rgb:* [0 0 0].
This can be used to raise the threshold on a specific color and thereby reduce the glows generated on areas of the source clip containing that color.

threshold: *Default:* 0.7, *Range:* 0 to 1.

Darkening will occur around locations in the source clip that are brighter than this value. A value of 0.9 causes dark glows from only the brightest spots. A value of 0 causes glows for every non−black area.

glowFromAlpha: *Default:* 0, *Range:* 0 to 1.
Set to 1 to generate glows from the alpha channel of the source input instead of the RGB channels. In this case the glows will not pick up color from the source and will typically be brighter. Values between 0 and 1 interpolate between using the RGB and the Alpha.

glowUnderSource: *Default:* 0, *Range:* 0 to 1.
Set to 1 to composite the Source input over the glows.

scaleSource: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the Source input when combined with the glows. This does not affect the generation of the glows themselves.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background. This parameter only has an effect if the background input is provided, and is visible due to a partially transparent Source image or a reduced Scale Source parameter value.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Glows(Source, Background, S_Mask, 1.06, "Glow Orthicon", darksWidth, etc...);`

See Also:

Glow
GlowDist
GlowRainbow
GlowAura
GlowRings
GlowDarks
GlowEdges
GlowNoise
Glint
Sapphire Plug−ins Introduction

GlowRainbow

In the S_Glows Plugin.

Generates rainbow colored glows based on the distances from the edges of the source input. Any edges in the input image, where the brightness crosses the given threshold value, will generate an equal glow into the darker side of the edges. This is best observed when used on images with dark backgrounds.

Inputs:

Source: The input clip that determines the glow locations and colors. This clip is also used as the background and the glows are layered back onto it.

Background: *Optional.* The clip to combine the glows with. If no background is given, the Source is also used as the Background.

S_Mask: *Optional.* If connected, the source glow colors are scaled by this input. A monochrome Mask can be used to choose a subset of Source areas that will generate glows. A color Mask can be used to selectively adjust the glow colors in different regions. The Mask is applied to the source before the glows are generated so it will not clip the resulting glows.

Parameters:

glowWidth: *Default:* .2*width, *Range:* 0 or greater.
Scales the glow distance. Note that a zero glow width still enhances the bright areas; set the brightness parameter to zero if you want to pass the Source through unchanged.

pixelAspect2: *Toggle−button, Default:* GetDefaultAspect()−.5.
Enable this to squash the pattern vertically by a factor of 2. This effect is not able to adjust for arbitrary pixel aspect ratios, but you can select between pixel aspects of 1 or 2.

frequency: *Default:* 12, *Range:* 0 or greater.
The frequency of the color pattern. Increase for more cycles through the spectrum.

frequencyRed: *Default:* 1, *Range:* 0 or greater.
Scales the red frequency.

frequencyGreen: *Default:* 0.9, *Range:* 0 or greater.
Scales the green frequency.

frequencyBlue: *Default:* 0.8, *Range:* 0 or greater.
Scales the blue frequency.

phase: *Default:* 0, *Range:* any.
Shifts the color pattern.

phaseSpeed: *Default:* 1, *Range:* any.
If non−zero, the color phase is automatically animated at this speed, causing the color pattern to flow over time.

phaseRed: *Default:* 0, *Range:* any.
Shifts the red phase.

phaseGreen: *Default:* 0, *Range:* any.
Shifts the green phase.

phaseBlue: *Default:* 0, *Range:* any.
Shifts the blue phase.

color: *Default rgb:* [1 1 1].
Scales the color of the glows.

brightness: *Default:* 0.8, *Range:* 0 or greater.
Scales the brightness of the glows.

glowSaturation: *Default:* 1, *Range:* any.
Scales the saturation of the glow colors. Increase for more intense colors. Set to 0 for monochrome.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the glows. The maximum of the red, green, and blue glow brightness is scaled by this value and combined with the background Alpha at each pixel.

thresholdAddColor: *Default rgb:* [0 0 0].
This can be used to raise the threshold on a specific color and thereby reduce the glows generated on areas of the source clip containing that color.

threshold: *Default:* 0.5, *Range:* 0 to 1.
Glows are generated at the edges of areas in the source clip that are brighter than this value. A value of 0.9 causes glows from only the brightest spots. A value of 0 causes glows for every non−black area.

glowFromAlpha: *Default:* 0, *Range:* 0 to 1.
Set to 1 to generate glows from the alpha channel of the source input instead of the RGB channels. In this case the glows will not pick up color from the source and will typically be brighter. Values between 0 and 1 interpolate between using the RGB and the Alpha.

glowUnderSource: *Default:* 0, *Range:* 0 to 1.
Set to 1 to composite the Source input over the glows.

lightBackground: *Default:* 0, *Range:* 0 to 1.
Increase this to give a look of the glow casting light onto the background image. To see this more clearly you can also lower the Scale Background parameter or raise the Brightness parameter.

scaleSource: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the Source input when combined with the glows. This does not affect the generation of the glows themselves.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background. This parameter only has an effect if the background input is provided, and is visible due to a partially transparent Source image or a reduced Scale Source parameter value.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Glows(Source, Background, S_Mask, 1.06, "Glow Rainbow", glowWidth, etc...);`

See Also:

Glow
GlowDist
GlowAura
GlowRings
GlowDarks
GlowOrthicon
GlowEdges
GlowNoise
Glint
PsykoStripes
PseudoColor
Sapphire Plug−ins Introduction

GlowRings

In the S_Glows Plugin.

Generates glows of colored rings around the areas of the source clip that are brighter than the given threshold, and then combines these with the source clip.

Inputs:

Source: The input clip that determines the glow locations and colors. This clip is also used as the background and the glows are layered back onto it.

Background: *Optional.* The clip to combine the glows with. If no background is given, the Source is also used as the Background.

S_Mask: *Optional.* If connected, the source glow colors are scaled by this input. A monochrome Mask can be used to choose a subset of Source areas that will generate glows. A color Mask can be used to selectively adjust the glow colors in different regions. The Mask is applied to the source before the glows are generated so it will not clip the resulting glows.

Parameters:

glowWidth: *Default:* .05*width, *Range:* 0 or greater.
Scales the glow distance. Note that a zero glow width still enhances the bright areas; set the brightness parameter to zero if you want to pass the Source through unchanged.

widthX: *Default:* 1, *Range:* 0 or greater.
Scales the horizontal glow width. Set to 0 for vertical only.

widthY: *Default:* 1/GetDefaultAspect(), *Range:* 0 or greater.
Scales the vertical glow width. Set to 0 for horizontal only.

widthRed: *Default:* 0.5, *Range:* 0 or greater.
Scales the red glow width. If the red, green, and blue widths are equal, the glows will match the color of the source clip. If they are not equal, the glows will vary in color with distance.

widthGreen: *Default:* 0.75, *Range:* 0 or greater.
Scales the green glow width.

widthBlue: *Default:* 1, *Range:* 0 or greater.
Scales the blue glow width.

subpixel: *Toggle−button, Default:* off.
Enables glowing by subpixel widths. Use this for smoother animation of the Width parameters.

thicknessRed: *Default:* 0.5, *Range:* 0 to 1.
Scales the thickness of the red region.

thicknessGreen: *Default:* 0.5, *Range:* 0 to 1.
Scales the thickness of the green region.

thicknessBlue: *Default:* 0.5, *Range:* 0 to 1.
Scales the thickness of the blue region.

color: *Default rgb:* [1 1 1].
Scales the color of the glows. The colors and brightnesses of the glows are also affected by the Source and Mask inputs.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of all the glows.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the glows. The maximum of the red, green, and blue glow brightness is scaled by this value and combined with the background Alpha at each pixel.

thresholdAddColor: *Default rgb:* [0 0 0].
This can be used to raise the threshold on a specific color and thereby reduce the glows generated on areas of the source clip containing that color.

threshold: *Default:* 0, *Range:* 0 to 1.
Glows are generated from locations in the source clip that are brighter than this value. A value of 0.9 causes glows at only the brightest spots. A value of 0 causes glows for every non−black area.

glowFromAlpha: *Default:* 0, *Range:* 0 to 1.
Set to 1 to generate glows from the alpha channel of the source input instead of the RGB channels. In this case the glows will not pick up color from the source and will typically be brighter. Values between 0 and 1 interpolate between using the RGB and the Alpha.

glowUnderSource: *Default:* 0, *Range:* 0 to 1.
Set to 1 to composite the Source input over the glows.

lightBackground: *Default:* 0, *Range:* 0 to 1.
Increase this to give a look of the glow casting light onto the background image. To see this more clearly you can also lower the Scale Background parameter or raise the Brightness parameter.

scaleSource: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the Source input when combined with the glows. This does not affect the generation of the glows themselves.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background. This parameter only has an effect if the background input is provided, and is visible due to a partially transparent Source image or a reduced Scale Source parameter value.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Glows(Source, Background, S_Mask, 1.06, "Glow Rings", glowWidth, etc...);`

See Also:

Glow
GlowDist
GlowRainbow
GlowAura
Glint
Sapphire Plug−ins Introduction

Gradient

Makes a smooth color gradient across the screen using given Start and End locations and colors, then optionally combines the gradient with a background clip. Increase Add Noise to reduce banding artifacts in the gradient due to color quantization.

Inputs:

Background: The clip to combine the gradient with.

Parameters:

start: *X & Y*, *Default:* [.5*width .9*height], *Range:* any.
The starting location of the gradient.

end: *X & Y*, *Default:* [.5*width .1*height], *Range:* any.
The ending location of the gradient.

startColor: *Default rgb:* [1 1 1].
The color of the gradient at the Start location.

endColor: *Default rgb:* [0 0 0.3].
The color of the gradient at the End location

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the gradient image (both the Start Color and End Color).

addNoise: *Default:* 0, *Range:* 0 or greater.
If positive, this amount of noise is added to the gradient. This can create a grainy effect and eliminate banding in the gradient due to quantization. Set this to 1.0 to enable effective debanding for 8bit results.

smoothCurve: *Default:* 0, *Range:* 0 to 1.
If zero, a linear interpolation is used across the screen between the Start and End Color. Increase this value to use a smoother 'S' shaped curve for interpolation which can reduce the visual perception of the gradient's Start and End locations.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background before combining it with the gradient.

combine: *Popup menu, Default: Grad Only*.
Determines how the gradient is combined with the background.

Grad Only: gives the gradient image alone with no background.
Mult: the background is multiplied by the gradient.
Add: the background is added to the gradient.
Screen: the background is blended with the gradient using a screen operation.
Difference: the result is the difference between the background and gradient.
Overlay: combines gradient and background using an overlay function.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Gradient(Background, 1.06, startX, etc...);`

See Also:

WipeLine
Sapphire Plug−ins Introduction

Grain

In the S_Grain Plugin.

Adds color and/or monochrome grain to the source clip. Amplitude and frequency parameters allow adjusting the grain texture independently for all colors together, each color channel, or black and white grain.

Inputs:

Source: The clip to be processed.

Parameters:

colorScale: *Default rgb:* [1 1 1].
Scales the color of the grain by this value. The grain will include both positive and negative values of this color.

colorAmplitude: *Default:* 0.1, *Range:* 0 or greater.
The amplitude of color grain to include.

colorFrequency: *Default:* 100, *Range:* 0.1 or greater.
The frequency of the color grain. Increase for finer color grain, decrease for coarser color grain.

redFreq: *Default:* 1, *Range:* 0.01 or greater.
The relative frequency of the red channel grain.

greenFreq: *Default:* 1, *Range:* 0.01 or greater.
The relative frequency of the green channel grain.

blueFreq: *Default:* 1, *Range:* 0.01 or greater.
The relative frequency of the blue channel grain.

colorOctaves: *Integer, Default:* 1, *Range:* 1 to 10.
The number of octaves of color grain to include. Each octave is twice the frequency and half the amplitude of the previous.

bwAmplitude: *Default:* 0, *Range:* 0 or greater.
The amplitude of black and white grain to include.

bwFrequency: *Default:* colorFrequency, *Range:* 0.1 or greater.
The frequency of the black and white grain. Increase for finer grain, decrease for coarser grain.

bwOctaves: *Integer, Default:* 1, *Range:* 1 to 10.
The number of octaves of black and white grain to include. Each octave is twice the frequency and half the amplitude of the previous.

seed: *Default:* 0.123, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

jitterFrames: *Integer, Default:* 1, *Range:* 0 or greater.
If this is 0, the noise texture will remain the same for every frame processed. If it is 1, a new noise texture is used for each frame. If it is 2, a new noise texture is used for every other frame, and so on.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Grain(Source, 1.06, "Grain", colorScaleRed, etc...);`

See Also:

Static
GrainRemove
FilmEffect
Diffuse
Clouds
DissolveSpeckle
Sapphire Plug−ins Introduction

Grain: Static

In the S_Grain Plugin.

Adds color and/or monochrome random noise of given amplitudes to every pixel of the source clip. Unlike the other Grain effects, there is no coherency of the grain between pixels, so the resulting look will vary with different output resolutions.

Inputs:

Source: The clip to be processed.

Parameters:

colorScale: *Default rgb:* [1 1 1].
Scales the color of the static by this value. The static will include both positive and negative values of this color.

colorAmplitude: *Default:* 0.1, *Range:* 0 or greater.
The amplitude of the color static to include.

bwAmplitude: *Default:* 0, *Range:* 0 or greater.
The amplitude of the black and white static to include.

Script Form: `S_Grain(Source, 1.06, "Static", colorScaleRed, etc...);`

See Also:

Grain
GrainRemove
FilmEffect
Diffuse
Clouds
DissolveSpeckle
Sapphire Plug−ins Introduction

GrainRemove

In the S_Grain Plugin.

Smooths the source clip while retaining the edges. To adjust the parameters in this effect, first use the Show:Edges option to inspect which edges will be retained and adjust Edges Threshold, Edges Width, and Edges Scale until the important edges are fairly sharp and bright but not jaggy. Then return to Show:Result and adjust the smooth parameters to remove the appropriate amount of grain.

Inputs:

Source: The clip to be processed.

Parameters:

smooth: *Default:* .01*width, *Range:* 0 or greater.
The amount of smoothing to apply to the non−edge regions.

smoothLuma: *Default:* 0.5, *Range:* 0 or greater.
Scales the smoothing amount for the luminance component.

smoothChroma: *Default:* 1, *Range:* 0 or greater.
Scales the smoothing amount for the chrominance component.

edgesWidth: *Default:* 2, *Range:* 0 or greater.
The width of the edges to be retained.

edgesScale: *Default:* 0.25, *Range:* 0 or greater.
The brightness of the edges to be retained.

edgesThreshold: *Default:* 0.3, *Range:* 0 or greater.
This value is subtracted from the initial edge image. Increasing it can help remove minor edges and speckles that should not be retained.

show: *Radio buttons, Default: Result.*
Selects between output options.

Result: outputs the final result.
Edges: outputs an image showing which edges are to be retained.

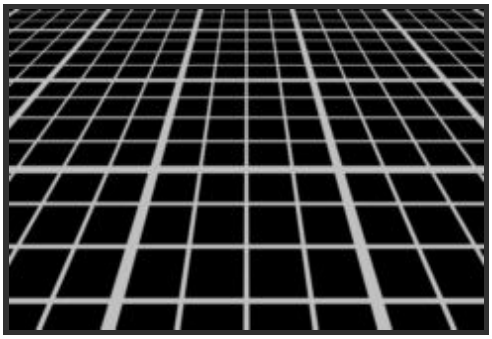
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Grain(Source, 1.06, "Grain Remove", smooth, etc...);`

See Also:

Grain
Blur
Static
Sharpen
Sapphire Plug−ins Introduction

Grid



Generates a grid of lines and combines it with a background clip. Adjust the Latitude, Swing, and Roll parameters to rotate the grid on various axes, and adjust Shift and Z Dist to translate and zoom.

Inputs:

Background: The clip to draw the grid on.

Parameters:

boxes: *X & Y, Integer, Default:* [24 16], *Range:* 1 to 200.
The total number of grid cells in the horizontal and vertical directions.

gridSize: *Default:* width, *Range:* 0 or greater.
Scales the size of the grid object.

gridSizeX: *Default:* 1, *Range:* 0 or greater.
Scales the relative horizontal size of the grid.

gridSizeY: *Default:* 0.75, *Range:* 0 or greater.
Scales the relative vertical size of the grid.

shift_: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translates the grid by this amount.

lineWidth: *Default:* 1, *Range:* 0 or greater.
Scales the thickness of all the grid lines.

hLineRelWidth: *Default:* 1, *Range:* 0 or greater.
Scales the relative thickness of the horizontal lines.

vLineRelWidth: *Default:* 1, *Range:* 0 or greater.
Scales the relative thickness of the vertical lines.

majorLineSpacing: *Integer, Default:* 4, *Range:* 0 to 200.
Thicker lines are drawn at each interval of this many lines. If zero, the major lines are disabled and all lines will be equal width.

majorLineWidth: *Default:* 2.5, *Range:* 1 or greater.
The relative thickness of the major lines.

color: *Default rgb:* [1 1 1].
The color of the grid.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the grid color.

opacity: *Default:* 1, *Range:* 0 to 1.
The opacity of the grid. Lower values allow more background to show through.

latitude: *Default:* 0, *Range:* any.
Tilts the grid up or down by this many degrees.

swing: *Default:* 0, *Range:* any.

Rotation of the grid in counter−clockwise degrees in its initial frame.

roll: *Default:* 0, *Range:* any.
Tilts the grid from side to side, in counter−clockwise degrees. If Latitude is 0, the effects of Swing and Roll are the same.

zDist: *Default:* 1, *Range:* 0.01 or greater.
Scales the 'distance' of the grid. Values greater than 1.0 move it farther away and make it smaller. Values less then 1.0 move it closer and enlarge it.

teleLensWidth: *Default:* 1, *Range:* 0.2 to 3.
The amount of lens telescoping. Increase to zoom in with less perspective, decrease for a wider viewing angle with more perspective.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background before combining with the grid. If 0, the result will contain only the grid image over black.

combine: *Radio buttons, Default: Over*.
Determines how the grid is combined with the Background.

> ***Over:*** composites the grid over the background.
> ***Exclusion:*** combines the grid and the Background with a difference operator.

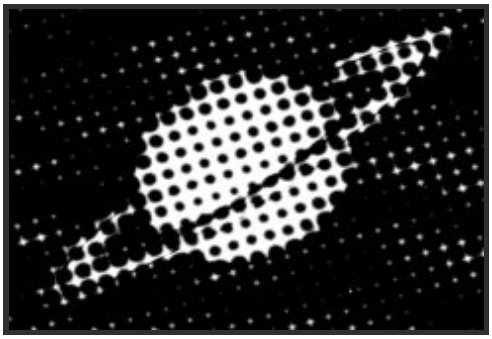
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Grid(Background, 1.06, boxesX, etc...);`

See Also:

WipeChecker
Sapphire Plug−ins Introduction

HalfTone



In the S_HalfTone Plugin.

Generates a duotone version of the source clip using a black and white pattern of dots. Use the Smooth Source parameter to remove some details and make the dots more consistently round.

Inputs:

Source: The clip to be processed.

Parameters:

dots: *Radio buttons, Default: Black.*
Selects the dots' color model.

Black: dark dots are used on a bright background.
White: bright dots are used on a dark background.

dotsFrequency: *Default:* 50, *Range:* 0 or greater.
The frequency of the dots pattern. Increase for finer dots, decrease for larger dots.

dotsAngle: *Default:* 30, *Range:* any.
The angle of the overall dots pattern, in counter−clockwise degrees.

dotsRelWidth: *Default:* 1, *Range:* 0.01 or greater.
The relative width of the dots. Increase for wider dots, decrease for taller ones.

dotsSharpness: *Default:* 4, *Range:* 0 or greater.
Scales the sharpness of the edges of the dots.

dotsLighten: *Default:* 0, *Range:* −1 to 1.
Increase to lighten the resulting dot pattern.

smoothSource: *Default:* 0, *Range:* 0 or greater.
If positive, the source is blurred by this amount before the halftone is applied. This can be used to remove some detail in the dots and make them more consistently round.

color1: *Default rgb:* [1 1 1].
The 'bright' color to use for the dots pattern.

color0: *Default rgb:* [0 0 0].
The 'dark' color to use for the dots pattern.

dotsShift: *X & Y, Default:* [0 0], *Range:* any.
The horizontal and vertical translation of the dots pattern

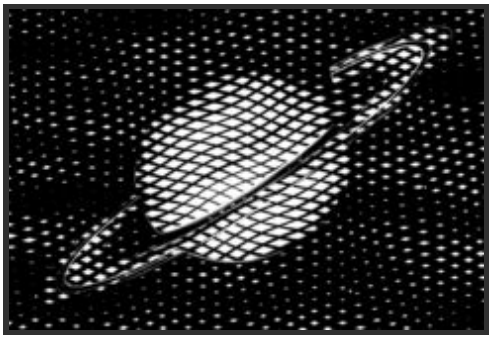
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_HalfTone(Source, 1.06, "Half Tone", dots, etc...);`

See Also:

HalfToneColor
HalfToneRings
Etching
ScanLines
WipeDots
VanGogh
Sketch
Mosaic
FlysEyeHex
JpegDamage
Sapphire Plug−ins Introduction

HalfTone: Etching



In the S_HalfTone Plugin.

Generates a version of the source clip using two sets of black and white lines of varying thickness to give an 'etching' or 'lithograph' look. Use the Smooth Source parameter to remove some details and make the lines more evenly shaped. Use the Lines Frequency parameter to adjust the density of all lines.

Inputs:

Source: The clip to be processed.

Parameters:

linesFrequency: *Default:* 50, *Range:* 0 or greater.
The frequency of the etched lines. Increase for a finer line pattern, decrease for fewer lines.

lines1Frequency: *Default:* 1, *Range:* 0 or greater.
Scales the frequency of the first set of etched lines. Increase for a finer line pattern, decrease for fewer lines.

lines2Frequency: *Default:* 1, *Range:* 0 or greater.
Scales the frequency of the second set of etched lines.

linesAngle: *Default:* 0, *Range:* any.
Rotation of the etched lines pattern in counter−clockwise degrees.

lines1Angle: *Default:* 30, *Range:* any.
The relative angle of the first set of etched lines in counter−clockwise degrees.

lines2Angle: *Default:* −20, *Range:* any.
The relative angle of the second set of etched lines in counter−clockwise degrees.

linesSharpness: *Default:* 4, *Range:* 0 or greater.
The sharpness of the etched lines. Decrease for softer edges.

linesAddWidth: *Default:* 0, *Range:* any.
Increase for thicker lines.

smoothSource: *Default:* 0, *Range:* 0 or greater.
If positive, the source is blurred by this amount before the etching is applied.

color1: *Default rgb:* [1 1 1].
The 'brighter' color of the lines pattern.

color0: *Default rgb:* [0 0 0].
The 'darker' color of the lines pattern.

waveAmp: *Default:* 0.1, *Range:* 0 or greater.
The amplitude of the waviness of the sets of etched lines.

waveFrequency: *Default:* 2, *Range:* 0 or greater.
The frequency of the waviness of the etched lines. Increase for more waves.

warpAmp: *Default:* 0.04, *Range:* any.
The amount the output is warped using the source brightness.

warpSmooth: *Default:* .005*width, *Range:* 0 or greater.
The smoothness of the warping. This has no effect if Warp Amp is 0.

edgesScale: *Default:* 0.5, *Range:* 0 or greater.
Adjusts the amount of source edges to be included in the result. If positive, edges in the source image are found and added to the etching pattern.

edgesThreshold: *Default:* 0.3, *Range:* 0 or greater.
Determines which edges are included in the result. Increase to remove minor edges and speckles. This has no effect unless Edges Scale is positive.

edgesWidth: *Default:* 0, *Range:* 0 or greater.
The width of the edges added to the result. Increase for wider edges. This has no effect unless Edges Scale is positive.

edgesSharpness: *Default:* 3, *Range:* 0 or greater.
Increase for sharper edges, decrease for softer edges. This has no effect unless Edges Scale is positive.

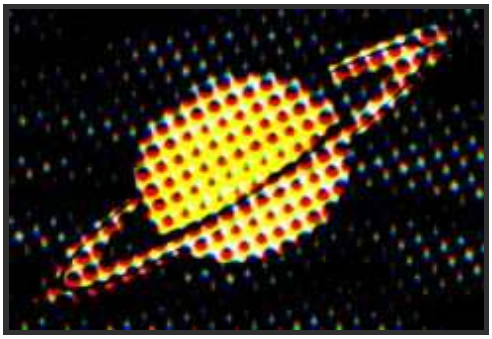
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_HalfTone(Source, 1.06, "Etching", linesFrequency, etc...);`

See Also:

ScanLines
VanGogh
Sketch
Mosaic
FlysEyeHex
JpegDamage
Sapphire Plug−ins Introduction

HalfToneColor



In the S_HalfTone Plugin.

Generates a version of the source clip using a colored dot pattern. Use the Smooth Source parameter to remove some details and make the dots more consistently round. You can invert the dots pattern from CMY to RGB using the Dots menu.

Inputs:

Source: The clip to be processed.

Parameters:

dotsColor: *Radio buttons, Default: CMY.*
Selects the dots' color model.

CMY: cyan, magenta, and yellow dots are used on a white background.
RGB: red, green, and blue dots are used on a black background.

dotsFrequency: *Default:* 50, *Range:* 0 or greater.
The frequency of the dots pattern. Increase for finer dots, decrease for larger dots.

dotsAngle: *Default:* 30, *Range:* any.
The angle of the overall dots pattern, in counter−clockwise degrees.

dotsRelWidth: *Default:* 1, *Range:* 0.01 or greater.
The relative width of the dots. Increase for wider dots, decrease for taller ones.

dotsSharpness: *Default:* 4, *Range:* 0 or greater.
Scales the sharpness of the edges of the dots.

dotsLighten: *Default:* 0, *Range:* −1 to 1.
Increase to lighten the resulting dot pattern.

smoothSource: *Default:* 0, *Range:* 0 or greater.
If positive, the source is blurred by this amount before the halftone is applied. This can be used to remove some detail in the dots and make them more consistently round.

saturation: *Default:* 1, *Range:* 0 or greater.
Scales the color saturation. Increase for more intense colors. Set to 0 for monochrome.

dotsShift: *X & Y, Default:* [0 0], *Range:* any.
The horizontal and vertical translation of the dots pattern

shiftRed: *X & Y, Default:* [0 0.5], *Range:* any.
The translation of the red color channel.

shiftGreen: *X & Y, Default:* [0 0], *Range:* any.
The translation of the green color channel.

shiftBlue: *X & Y, Default:* [0 −0.5], *Range:* any.
The translation of the blue color channel.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111

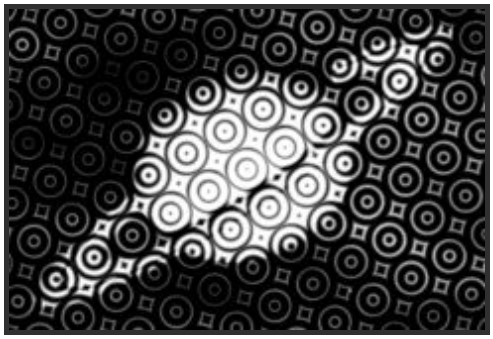
for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_HalfTone(Source, 1.06, "Half Tone Color", dotsColor, etc...);`

See Also:

HalfTone
HalfToneRings
Etching
ScanLines
WipeDots
VanGogh
Sketch
Mosaic
FlysEyeHex
JpegDamage
Sapphire Plug−ins Introduction

HalfToneRings



In the S_HalfTone Plugin.

Generates a duotone version of the source clip using a repeating pattern of concentric rings. Use the Smooth Source parameter to remove some details and make the dots more consistently shaped.

Inputs:

Source: The clip to be processed.

Parameters:

center: *Radio buttons, Default: Black.*
Selects the rings' color model.

Black: dark rings are used on a bright background.
White: bright rings are used on a dark background.

ringsFrequency: *Default:* 20, *Range:* 0 or greater.
The frequency of the overall rings pattern. Increase for smaller rings, decrease for larger rings.

ringsAngle: *Default:* 30, *Range:* any.
The angle of the overall rings pattern, in counter−clockwise degrees.

ringsRelWidth: *Default:* 1, *Range:* 0.01 or greater.
The relative width of the rings. Increase for wider rings, decrease for taller ones.

ringsSharpness: *Default:* 2, *Range:* −1 or greater.
Scales the sharpness of the edges of the rings.

ringsLighten: *Default:* 0, *Range:* −1 to 1.
Increase to lighten the resulting rings pattern.

ringNumber: *Default:* 2, *Range:* 1 or greater.
Determines the number of concentric rings in each tile of the repeating pattern.

ringPhase: *Default:* 0, *Range:* any.
Shifts the rings in or out within each tile of the pattern.

smoothSource: *Default:* 0, *Range:* 0 or greater.
If positive, the source is blurred by this amount before the halftone is applied. This can be used to remove some detail in the dots and make them more consistently round.

color1: *Default rgb:* [1 1 1].
The 'bright' color to use for the dots pattern.

color0: *Default rgb:* [0 0 0].
The 'dark' color to use for the dots pattern.

ringsShift: *X & Y, Default:* [0 0], *Range:* any.
The horizontal and vertical translation of the overall rings pattern

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_HalfTone(Source, 1.06, "Half Tone Rings", center, etc...);`

See Also:

HalfTone
HalfToneColor
Etching
ScanLines
WipeDots
VanGogh
Sketch
Mosaic
FlysEyeHex
JpegDamage
Sapphire Plug−ins Introduction

JpegDamage

Creates a version of the Source input that is subjected to Jpeg compression artifacts and errors. This can be used to give various looks of low quality digital transmissions. Three methods for manipulating your image are provided: the Jpeg quality can be adjusted, various internal frequencies can be scaled, and random decompression errors can be introduced. Note that Shake's proxyScale and interactiveScale can affect the results of this effect unless the resFactorX and Y parameters are set high enough.

Inputs:

Source: The clip to be processed.

Parameters:

quality: *Default:* 0.1, *Range:* 0.01 to 1.
Determines the amount of normal Jpeg artifacts. Use lower values for more compression.

resFactor: *X & Y, Integer, Default:* [1 resFactorX], *Range:* 1 or greater.
Downres the result by the inverse of this amount, so 1 is full resolution, 2 is 1/2, 3 is 1/3, etc. The jpeg block shapes will be rectangular if the x and y res factors are different from each other. You won't notice the result of this parameter unless its value is beyond then the current viewing downres factor.

allFreqScale: *Default:* 1, *Range:* 0 or greater.
Scales the frequencies for all Jpeg coefficients. Values other than 1 cause abnormal results, and create unusual looking blocky versions of your input.

xFreqScale: *Default:* 1, *Range:* 0 or greater.
Scales the horizontal Jpeg frequencies. Values other than 1 cause abnormal results.

yFreqScale: *Default:* 1, *Range:* 0 or greater.
Scales the vertical Jpeg frequencies.

lowFreqScale: *Default:* 1, *Range:* 0 or greater.
Scales the softer low frequencies.

midFreqScale: *Default:* 1, *Range:* 0 or greater.
Scales the middle range frequencies.

highFreqScale: *Default:* 1, *Range:* 0 or greater.
Scales the sharper high frequencies. You may need a high Quality setting to see the high frequencies at all.

affectLuma: *Default:* 1, *Range:* 0 or greater.
Determines how much the Freq Scale parameters above affect the luminance channel. A zero value causes no luminance change. Values greater than 1.0 exaggerate the change.

affectChroma: *Default:* 0.5, *Range:* 0 or greater.
Determines how much the Freq Scale parameters above affect the chroma channels. A zero value causes no chroma change. Values greater than 1.0 exaggerate the change.

errorRate: *Default:* 0, *Range:* 0 or greater.
If positive, random decompression errors are introduced. The value determines the average number of errors in those blocks that receive errors. Larger values give a more even grainy look.

errBlockDensity: *Default:* 0.75, *Range:* 0 to 1.
Determines the percentage of Jpeg blocks with errors. A value of .5 will give errors in half of the blocks and 1.0

will give errors in all blocks.

errorAmp: *Default:* 1, *Range:* 0 or greater.
The amplitude of the decompression errors. Larger values give more visually obvious errors. This has no effect unless the Error Rate is also positive.

errorCoherence: *Default:* 1, *Range:* 0 or greater.
Determines how much the blocks with errors are grouped together. When zero, the errors are evenly distributed throughout the frame. When increased, the errors are clustered into larger groups. This has no effect unless the Error Rate is positive and the Err Block Density is less than 1.

jitterFrames: *Integer*, *Default:* 1, *Range:* 0 or greater.
If this is 0, the random errors will remain the same for every frame processed. If it is 1, different errors are used for each frame. If it is 2, new errors are used for every other frame, and so on. This has no effect unless the Error Rate is also positive.

randSeed: *Default:* 0.123, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different random error patterns, and the same value should give a repeatable result. This has no effect unless the Error Rate is also positive.

scaleLights: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result by this amount.

offsetDarks: *Default:* 0, *Range:* any.
Adds this gray value to the darker regions of the source. This can be negative to increase contrast.

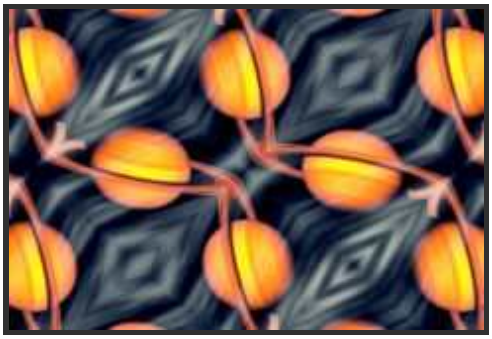
saturation: *Default:* 1, *Range:* any.
Scales the color saturation. Increase for more intense colors. Set to 0 for monochrome.

Script Form: `S_JpegDamage(Source, 1.06, quality, etc...);`

See Also:

ScanLines
HalfTone
Mosaic
FlysEyeRect
VanGogh
Sapphire Plug−ins Introduction

KaleidoDiamonds



In the S_Kaleido Plugin.

Reflects the source clip into a pattern of diamonds. The 'Inside' parameters transform the Source image before it is reflected into the pattern. The Center and Z Dist transform the entire result including the reflection pattern, and the Rotate affects only the reflecting 'mirrors'.

Inputs:

Source: The clip to be processed.

Parameters:

center: *X & Y*, *Default:* [.5*width .5*height], *Range:* any.
Center location of the kaleidoscoped image in screen coordinates relative to the center of the frame. The entire result will be shifted by this amount.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the entire result in or out from the Center. Increase to zoom out, decrease to zoom in.

rotate: *Default:* 0, *Range:* any.
Rotates the kaleidoscope's reflection pattern about the Center by this many counter−clockwise degrees.

insideShift: *X & Y*, *Default:* [0 0], *Range:* any.
Translates the source image inside the kaleidoscope before it is reflected.

insideZDist: *Default:* 1, *Range:* 0.001 or greater.
Zooms the source image in or out inside the kaleidoscope before it is reflected.

insideRotate: *Default:* 0, *Range:* any.
Rotates the source image inside the kaleidoscope before it is reflected.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

wrap: *X & Y*, *Radio buttons*, *Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image. This is used only if the image inside the kaleidoscope is not contained within the shape of mirrors.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button*, *Default:* on.
If enabled, the Source image is resampled using pixel averaging. This removes aliasing and gives a higher quality result, although it may not be necessary if your input image is smooth with no sharp edges or high frequencies.

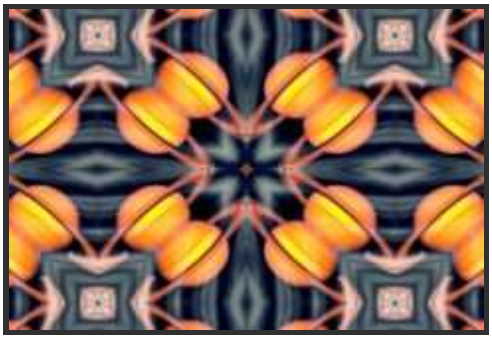
Script Form: `S_Kaleido(Source, 1.06, "Kaleido Diamonds", centerX, etc...);`

See Also:

KaleidoTriangles
KaleidoSquares
KaleidoOct
KaleidoPolar

Sapphire Plug−ins Introduction

KaleidoOct



In the S_Kaleido Plugin.

Reflects the source clip into an octagonal pattern of right triangles. The 'Inside' parameters transform the Source image before it is reflected into the pattern. The Center and Z Dist transform the entire result including the reflection pattern, and the Rotate affects only the reflecting 'mirrors'.

Inputs:

Source: The clip to be processed.

Parameters:

center: *X & Y*, *Default:* [.5*width .5*height], *Range:* any.
Center location of the kaleidoscoped image in screen coordinates relative to the center of the frame. The entire result will be shifted by this amount.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the entire result in or out from the Center. Increase to zoom out, decrease to zoom in.

rotate: *Default:* 0, *Range:* any.
Rotates the kaleidoscope's reflection pattern about the Center by this many counter−clockwise degrees.

insideShift: *X & Y*, *Default:* [0 0], *Range:* any.
Translates the source image inside the kaleidoscope before it is reflected.

insideZDist: *Default:* 1, *Range:* 0.001 or greater.
Zooms the source image in or out inside the kaleidoscope before it is reflected.

insideRotate: *Default:* 0, *Range:* any.
Rotates the source image inside the kaleidoscope before it is reflected.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

wrap: *X & Y*, *Radio buttons*, *Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image. This is used only if the image inside the kaleidoscope is not contained within the shape of mirrors.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button*, *Default:* on.
If enabled, the Source image is resampled using pixel averaging. This removes aliasing and gives a higher quality result, although it may not be necessary if your input image is smooth with no sharp edges or high frequencies.

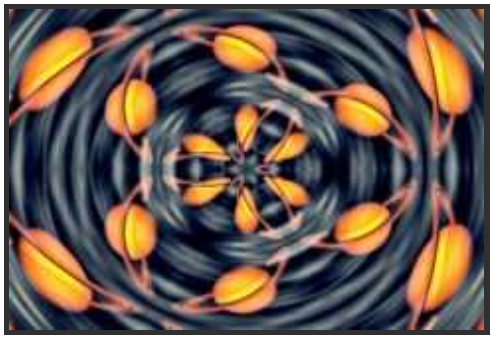
Script Form: `S_Kaleido(Source, 1.06, "Kaleido Oct", centerX, etc...);`

See Also:

KaleidoTriangles
KaleidoSquares
KaleidoDiamonds
KaleidoPolar

Sapphire Plug−ins Introduction

KaleidoPolar



In the S_Kaleido Plugin.

Warps the source clip around in a disk shape and reflects radially as if viewed through a reflecting cylinder.

Inputs:

Source: The input clip to be warped.

Parameters:

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Center location of the kaleidoscoped image in screen coordinates relative to the center of the frame. The entire result will be shifted by this amount.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the entire result in or out from the Center. Increase to zoom out, decrease to zoom in.

rotate: *Default:* 0, *Range:* any.
Rotates the kaleidoscope's reflection pattern about the Center by this many counter−clockwise degrees.

stretch: *X & Y, Default:* [1 1], *Range:* 0.1 or greater.
Scales the horizontal or vertical size of the result.

insideShiftY: *Default:* 0, *Range:* any.
Shifts the source image up by this amount before it is reflected. This causes the resulting pattern of images to radiate outward from the center.

angleRepeats: *Default:* 6, *Range:* 0.01 or greater.
The number of copies of the source image to wrap around. This should be an even integer to avoid a seam where the first copy connects to the last.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

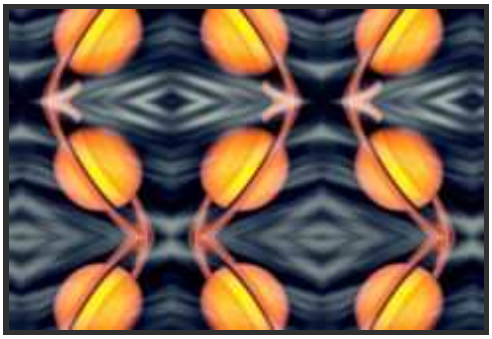
filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

Script Form: `S_Kaleido(Source, 1.06, "Kaleido Polar", centerX, etc...);`

See Also:

KaleidoTriangles
KaleidoSquares
KaleidoDiamonds
KaleidoOct
WarpPolar
Sapphire Plug−ins Introduction

KaleidoSquares



In the S_Kaleido Plugin.

Reflects the source clip into a pattern of squares. The 'Inside' parameters transform the Source image before it is reflected into the pattern. The Center and Z Dist transform the entire result including the reflection pattern, and the Rotate affects only the reflecting 'mirrors'.

Inputs:

Source: The clip to be processed.

Parameters:

center: *X & Y*, *Default:* [.5*width .5*height], *Range:* any.
Center location of the kaleidoscoped image in screen coordinates relative to the center of the frame. The entire result will be shifted by this amount.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the entire result in or out from the Center. Increase to zoom out, decrease to zoom in.

rotate: *Default:* 0, *Range:* any.
Rotates the kaleidoscope's reflection pattern about the Center by this many counter−clockwise degrees.

insideShift: *X & Y*, *Default:* [0 0], *Range:* any.
Translates the source image inside the kaleidoscope before it is reflected.

insideZDist: *Default:* 1, *Range:* 0.001 or greater.
Zooms the source image in or out inside the kaleidoscope before it is reflected.

insideRotate: *Default:* 0, *Range:* any.
Rotates the source image inside the kaleidoscope before it is reflected.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

wrap: *X & Y*, *Radio buttons*, *Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image. This is used only if the image inside the kaleidoscope is not contained within the shape of mirrors.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button*, *Default:* on.
If enabled, the Source image is resampled using pixel averaging. This removes aliasing and gives a higher quality result, although it may not be necessary if your input image is smooth with no sharp edges or high frequencies.

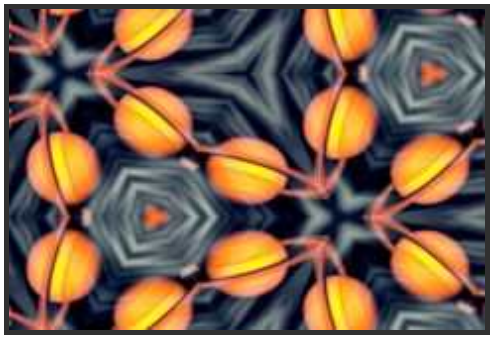
Script Form: `S_Kaleido(Source, 1.06, "Kaleido Squares", centerX, etc...);`

See Also:

KaleidoTriangles
KaleidoDiamonds
KaleidoOct
KaleidoPolar

Sapphire Plug−ins Introduction

KaleidoTriangles



In the S_Kaleido Plugin.

Reflects the source clip into a pattern of equilateral triangles. The 'Inside' parameters transform the Source image before it is reflected into the pattern. The Center and Z Dist transform the entire result including the reflection pattern, and the Rotate affects only the reflecting 'mirrors'.

Inputs:

Source: The clip to be processed.

Parameters:

center: *X & Y*, *Default:* [.5*width .5*height], *Range:* any.
Center location of the kaleidoscoped image in screen coordinates relative to the center of the frame. The entire result will be shifted by this amount.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the entire result in or out from the Center. Increase to zoom out, decrease to zoom in.

rotate: *Default:* 0, *Range:* any.
Rotates the kaleidoscope's reflection pattern about the Center by this many counter−clockwise degrees.

insideShift: *X & Y*, *Default:* [0 0], *Range:* any.
Translates the source image inside the kaleidoscope before it is reflected.

insideZDist: *Default:* 1, *Range:* 0.001 or greater.
Zooms the source image in or out inside the kaleidoscope before it is reflected.

insideRotate: *Default:* 0, *Range:* any.
Rotates the source image inside the kaleidoscope before it is reflected.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

wrap: *X & Y*, *Radio buttons*, *Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image. This is used only if the image inside the kaleidoscope is not contained within the shape of mirrors.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button*, *Default:* on.
If enabled, the Source image is resampled using pixel averaging. This removes aliasing and gives a higher quality result, although it may not be necessary if your input image is smooth with no sharp edges or high frequencies.

Script Form: `S_Kaleido(Source, 1.06, "Kaleido Triangles", centerX, etc...);`

See Also:

KaleidoSquares
KaleidoDiamonds
KaleidoOct
KaleidoPolar

Sapphire Plug−ins Introduction

Layer

Layers the Foreground image over the Background using one of a variety of blending operations. The colors of each input can also be adjusted using the lights, darks, and saturation parameters.

Inputs:

Foreground: The clip to use as foreground.

Background: The clip to use as background.

Matte: *Optional.* Specifies the opacities of the Foreground clip. If this input is not provided the Foreground is assumed fully opaque. These values are scaled by the Opacity parameter before being used.

Parameters:

mode: *Popup menu, Default: Normal.*
Determines which blending method is used to combine the foreground and background pixel colors.

Normal: a normal composite. This will just give the foreground as the result unless the Opacity is below 1.0 or an alpha channel is given.
Dissolve: randomly replaces background pixels with foreground. The opacity determines the probability, so the foreground is more likely to replace the background for higher values of Opacity.
Multiply: this can be used as an 'intersection' operation on matte images. White is the identity for Multiply, where one image contains white the other is not affected, so the result only contains white where both inputs are white.
Screen: this can be useful for combining the bright areas of two clips. It can also be used as a 'union' operation on matte images. Black is the identity for Screen, where one image contains black the other is not affected, so the result is white where either of the input images is white.
Overlay: combines foreground and background using an overlay function.
Soft Light: darkens or lightens the background depending on the foreground.
Hard Light: similar to Overlay but with foreground and background swapped.
Color Dodge: brightens the background depending on the foreground.
Color Burn: darkens the background depending on the foreground.
Darken: the minimum of foreground and background. This can also be used as an 'intersection' operation with slightly different results than Multiply.
Lighten: the maximum of foreground and background. This can also be used as a 'union' operation with slightly different results than Screen.
Add: adds the foreground to the background.
Subtract: subtracts the foreground from the background.
Difference: similar to Subtract but the absolute value of the result is used, which tends to give more resulting colors in bounds. This can be used to select the regions of two matte images where one or the other is white, but not both.
Exclusion: similar to Difference but with smoother results.
Hue: combines the hue of the foreground with the saturation and luminance of the background.
Saturation: combines the saturation of the foreground with the hue and luminance of the background.
Chroma: combines the hue and saturation of the foreground with the luminance of the background.
Luminance: combines the luminance of the foreground with the hue and saturation of the background.

compPremult: *Toggle−button, Default:* on.
Enable this for a better composite if the Foreground pixel values have been pre−multiplied by the original Matte pixel values. This will give an 'additive' composite when in Normal mode. This has no effect if Use Alpha is No.

useAlpha: *Toggle−button, Default:* on.

Selects how to use the Alpha channel of the Foreground to composite over the Background.

swapInputs: *Toggle−button, Default:* off.
If enabled, effectively swaps the Background and Foreground inputs, and can be helpful for non−commutative operations like subtract. Note that this also causes parameters labeled 'Front' to affect the 'Back' input instead, and vice versa.

fgOpacity: *Default:* 1, *Range:* 0 to 1.
Scales the opacity of the effect. When this is decreased the result approaches the background. At zero, the result will equal the background.

fgLights: *Default:* 1, *Range:* 0 or greater.
Scales the Foreground before performing the effect.

fgDarks: *Default:* 0, *Range:* any.
Offsets the darker regions of the Foreground before performing the effect. This can be negative to increase contrast.

fgSaturation: *Default:* 1, *Range:* 0 or greater.
Scales the color saturation of the Foreground before performing the effect. 0.0 makes it monochromatic, 1.0 has no effect.

bgLights: *Default:* 1, *Range:* 0 or greater.
Scales the Background before performing the effect.

bgDarks: *Default:* 0, *Range:* any.
Offsets the darker regions of the Background before performing the effect. This can be negative to increase contrast.

bgSaturation: *Default:* 1, *Range:* 0 or greater.
Scales the color saturation of the Background before performing the effect. 0.0 makes it monochromatic, 1.0 has no effect.

resultLights: *Default:* 1, *Range:* 0 or greater.
Scales the result after performing the effect.

resultDarks: *Default:* 0, *Range:* any.
Offsets the darker regions of the result after performing the effect. This can be negative to increase contrast.

resultSaturation: *Default:* 1, *Range:* 0 or greater.
Scales the color saturation of the result after performing the effect. 0.0 makes it monochromatic, 1.0 has no effect.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Layer(Foreground, Background, Matte, 1.06, "Normal", compPremult, etc...);`

See Also:

MathOps
Sapphire Plug−ins Introduction

LensFlare

In the S_LensFlare Plugin.

Renders a lens flare image over the background clip, aligning various flare elements between the hotspot and pivot locations. Use the Lens menu to select different types of lensflares.

Inputs:

Background: The clip to apply the lens flare over.

Track: *Optional*. This is only used if the AutoTrack effect option is also selected. The lensflare hotspot is positioned on the brightest part of this image. If this input is not enabled, the Background is used instead.

Parameters:

lens: *Popup menu, Default: 50_300mm_zoom*.
The type of lens flare to apply. Custom lens flare types can also be made, or existing types modified, by editing the "s_lensflares.text" file.

50_300mm_zoom: hotspot with ray cluster, red glow and red ring, and many other elements.
35mm_prime: hotspot with ray cluster, red glow and red ring, and a few other elements.
105mm_prime: hotspot with ray cluster, blue ring, blue glow, and other elements.
anamorphic1: horizontal blue rays, diagonal blue ray, red glow and red ring.
anamorphic2: horizontal blue rays and red glow.
anamorphicBlue: horizontal and diagonal blue rays with hexagonal blue/white glow.
rays_only: ray cluster with no other flare elements.
rays_only2: ray cluster like 105mm_prime, but with no other flare elements.
rays_only3: ray cluster like anamorphic1, but with no other flare elements.
simple_hex: four simple red and blue hexagon elements.

scaleWidths: *Default:* .2*width, *Range:* 0 or greater.
Scales the sizes of all the flare elements.

relHeights: *Default:* 1, *Range:* 0 or greater.
Scales the vertical dimension of all the flare elements, making them elliptical instead of circular.

raysRotate: *Default:* 0, *Range:* any.
Rotates the ray elements of the lens flare, if any, in counter−clockwise degrees.

hotspot: *X & Y, Default:* [.25*width .75*height], *Range:* any.
The location of the brightest spot in the flare in screen coordinates. The hotspot parameter is ignored if the AutoTrack effect option is selected.

pivot: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The elements of the flare will be in a line between the Hotspot and the Pivot locations. The Pivot location is in screen coordinates.

color: *Default rgb:* [1 1 1].
Scales the color of all flare elements.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of all the flare elements.

gamma: *Default:* 1, *Range:* 0 or greater.
Increasing gamma brightens the flare, and especially boosts the darker elements.

saturation: *Default:* 1, *Range:* any.
Scales the color saturation of the flare elements. Increase for more intense colors. Set to 0 for a monochrome lens flare.

hotspotColor: *Default rgb:* [1 1 1].
Scales the color of the hotspot elements only.

hotspotBright: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the hotspot elements only.

raysBrightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the ray elements only.

otherBrightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of all flare elements that are NOT at the hotspot location.

blurFlare: *Default:* 0, *Range:* 0 or greater.
If positive, the flare image is blurred by this amount before being combined with the background.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background before combining with the flare. If 0, the result will contain only the flare image over black.

combine: *Radio buttons, Default: Screen.*
Determines how the flare image is combined with the Background.

> ***Screen:*** performs a blend function which can help prevent overly bright results.
> ***Add:*** causes the flare image to be added to the background.

tintBgWhites: *Toggle−button, Default:* off.
If this is enabled, the chroma of the flare is added only after the result is clamped to the maximum brightness. This allows the color of the flare image to still be visible even over bright white backgrounds. For the majority of backgrounds there will be no observable difference.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the flare. The maximum of the red, green, and blue flare brightness is scaled by this value and combined with the Background Alpha at each pixel.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_LensFlare(Background, Track, 1.06, 0, lens, etc...);`

See Also:

LensFlareAutoTrack
Glint
Glare
Sapphire Plug−ins Introduction

LensFlareAutoTrack

In the S_LensFlare Plugin.

Renders a lens flare image over the background clip, aligning various flare elements between the hotspot and pivot locations. In this AutoTrack version of LensFlare, the hotspot is automatically positioned on the brightest area of the background clip. If there are multiple areas with the same maximum brightness, their locations are averaged. Increasing Blur For Auto will cause the input to be smoothed before the brightest location is found and can help remove the effect of secondary bright spots.

Inputs:

Background: The clip to apply the lens flare over.

Track: *Optional.* This is only used if the AutoTrack effect option is also selected. The lensflare hotspot is positioned on the brightest part of this image. If this input is not enabled, the Background is used instead.

Parameters:

lens: *Popup menu, Default: 50_300mm_zoom.*
The type of lens flare to apply. Custom lens flare types can also be made, or existing types modified, by editing the "s_lensflares.text" file.

50_300mm_zoom: hotspot with ray cluster, red glow and red ring, and many other elements.
35mm_prime: hotspot with ray cluster, red glow and red ring, and a few other elements.
105mm_prime: hotspot with ray cluster, blue ring, blue glow, and other elements.
anamorphic1: horizontal blue rays, diagonal blue ray, red glow and red ring.
anamorphic2: horizontal blue rays and red glow.
anamorphicBlue: horizontal and diagonal blue rays with hexagonal blue/white glow.
rays_only: ray cluster with no other flare elements.
rays_only2: ray cluster like 105mm_prime, but with no other flare elements.
rays_only3: ray cluster like anamorphic1, but with no other flare elements.
simple_hex: four simple red and blue hexagon elements.

scaleWidths: *Default:* .2*width, *Range:* 0 or greater.
Scales the sizes of all the flare elements.

relHeights: *Default:* 1, *Range:* 0 or greater.
Scales the vertical dimension of all the flare elements, making them elliptical instead of circular.

raysRotate: *Default:* 0, *Range:* any.
Rotates the ray elements of the lens flare, if any, in counter−clockwise degrees.

hotspotShift: *X & Y, Default:* [0 0], *Range:* any.
Adds this amount to the hotspot location. This allows for a relative adjustment of the hotspot away from the auto−tracked location if necessary.

pivot: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The elements of the flare will be in a line between the Hotspot and the Pivot locations. The Pivot location is in screen coordinates.

blurForAuto: *Default:* .01*width, *Range:* 0 or greater.
Used only if the effect is set to AutoTrack. The input is blurred by this amount before finding the brightest location.

color: *Default rgb:* [1 1 1].
Scales the color of all flare elements.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of all the flare elements.

gamma: *Default:* 1, *Range:* 0 or greater.
Increasing gamma brightens the flare, and especially boosts the darker elements.

saturation: *Default:* 1, *Range:* any.
Scales the color saturation of the flare elements. Increase for more intense colors. Set to 0 for a monochrome lens flare.

hotspotColor: *Default rgb:* [1 1 1].
Scales the color of the hotspot elements only.

hotspotBright: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the hotspot elements only.

raysBrightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the ray elements only.

otherBrightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of all flare elements that are NOT at the hotspot location.

blurFlare: *Default:* 0, *Range:* 0 or greater.
If positive, the flare image is blurred by this amount before being combined with the background.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background before combining with the flare. If 0, the result will contain only the flare image over black.

combine: *Radio buttons, Default: Screen.*
Determines how the flare image is combined with the Background.

> ***Screen:*** performs a blend function which can help prevent overly bright results.
> ***Add:*** causes the flare image to be added to the background.

tintBgWhites: *Toggle−button, Default:* off.
If this is enabled, the chroma of the flare is added only after the result is clamped to the maximum brightness. This allows the color of the flare image to still be visible even over bright white backgrounds. For the majority of backgrounds there will be no observable difference.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the flare. The maximum of the red, green, and blue flare brightness is scaled by this value and combined with the Background Alpha at each pixel.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_LensFlare(Background, Track, 1.06, 1, lens, etc...);`

See Also:

LensFlare
Glint
Glare
Sapphire Plug−ins Introduction

MathOps

Combines two clips using one of a variety of mathematical operations.

Inputs:

SourceA: The first input clip to be processed.

SourceB: The second input clip to be processed.

Parameters:

operation: *Popup menu, Default: Add.*
Determines which mathematical operation is applied to combine the pixel colors of the two source inputs.

Add: A + B.
Subtract: A − B.
Multiply: A * B. This can be used as an 'intersection' operation on matte images. The result only contains white where both inputs are white.
Divide: A / B. This can be used to 'un−premultiply' an image by using its matte as the second input.
Screen: A + B − AB. This can be useful in combining the bright areas of two clips. It can also be used as a 'union' operation on matte images. The result is white where either of the input images is white.
Average: (A + B) / 2.
Overlay: combines A over B with an overlay function.
Minimum: the smallest value for each color channel of each pixel. This can also be used as an 'intersection' operation with slightly different results than Multiply.
Maximum: the largest value for each color channel of each pixel. This can also be used as a 'union' operation with slightly different results than Screen.
Difference: similar to Subtract but the absolute value of the result is used, which tends to give more resulting colors in bounds. This can be used to select the regions of two matte images where one or the other is white, but not both.
Xor: performs an 'exclusive−or' operation on the colors of the source clips. This can also be used to select the regions of two matte images where one or the other is white, but not both, with slightly different results than Difference.
Xor Bits: performs a bitwise exclusive−or on the colors of the source clips. This can produce some interesting contour effects although the results are often difficult to predict.
And Bits: performs a bitwise logical and on the colors of the source clips. Similar to XorBits but tends to produce darker results.
Or Bits: performs a bitwise logical or on the colors of the source clips. Similar to XorBits but tends to produce brighter results.
Mod: gives the remainder after dividing the colors of the first source clip by the second. Set the A Scale parameter to a high value for some unusual pixel banding effects.
Round: the colors of the first source clip are rounded using the values of the second input as the step size.
Bounce: similar to Mod but the result contains fewer jagged edges. Set the A Scale parameter to a high value for some striping effects.

aLights: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of SourceA before performing the operation.

aDarks: *Default:* 0, *Range:* any.
Offsets the darker regions of the first input before performing the effect. This can be negative to increase contrast.

aSaturation: *Default:* 1, *Range:* 0 or greater.
Adjusts the color intensity of SourceA before performing the operation. 0.0 makes it monochromatic, 1.0 has no effect.

bLights: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of SourceB before performing the operation.

bDarks: *Default:* 0, *Range:* any.
Offsets the darker regions of the second input before performing the effect. This can be negative to increase contrast.

bSaturation: *Default:* 1, *Range:* 0 or greater.
Adjusts the color intensity of SourceB before performing the operation. 0.0 makes it monochromatic, 1.0 has no effect.

destLights: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result after performing the operation.

destDarks: *Default:* 0, *Range:* any.
Offsets the darker regions of the result after performing the effect. This can be negative to increase contrast.

destSaturation: *Default:* 1, *Range:* 0 or greater.
Scales the color intensity of the result after performing the operation. 0.0 makes it monochromatic, 1.0 has no effect.

swapInputs: *Toggle−button*, *Default:* off.
If enabled, effectively swaps the A and B Source inputs, and can be helpful for non−commutative operations like subtract. Note that this also causes parameters labeled 'A' to affect the 'B' input instead, and vice versa.

clipMode: *Radio buttons*, *Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_MathOps(SourceA, SourceB, 1.06, "Add", aLights, etc...);`

See Also:

Layer
Sapphire Plug−ins Introduction

MatteOps

Grows, shrinks, or adds noise to the alpha channel of the Source input. This can be useful for removing blue or green spill from a chroma key.

Inputs:

Source: The matte input clip to process. It is assumed to have anti−aliased but hard edges, because very soft edges might not be affected in a useful way. Only the alpha channel of this input is used (or green/luma if there is no alpha).

Parameters:

shrinkGrow: *Default:* 0, *Range:* any.
Amount to grow the matte edges in approximate pixels, or shrink if negative.

edgeSoftness: *Default:* 1, *Range:* 0.01 or greater.
The resulting softness of the edges.

postBlur: *Default:* 0, *Range:* 0 or greater.
If positive, the result is blurred by this amount. This is an alternative method for softening the edges.

filter: *Radio buttons, Default: Triangle*.
The type of blur filter to use for the shrink or grow process.

Box: uses a rectangular shaped filter.
Triangle: smoother, uses a pyramid shaped filter.
Gauss: smoothest, uses a gaussian shaped filter.

invertMatte: *Toggle−button, Default:* off.
If enabled, the black and white of the output matte are inverted.

output: *Radio buttons, Default: Matte*.
Selects the format of the output. If an RGBA type of output is selected, the input should also be RGBA.

Matte: the processed Matte is output on all channels.
RGBA: the Alpha output channel receives the processed Matte, and the RGB channels are passed through from the input unchanged. The input must also be RGBA for this to output all 4 channels.
RGBA Premult: the Alpha output channel receives the processed Matte, and the RGB channels receive the input multiplied by the new Matte. This option can be appropriate if you are shrinking a matte and need an RGBA result for pre−multiplied compositing. The input must also be RGBA for this to output all 4 channels.

noiseAmplitude: *Default:* 0, *Range:* 0 or greater.
The amount of noise texture to add to the edges.

noiseWidth: *Default:* 2, *Range:* 0 or greater.
The width of the area at the matte edges where the noise is included. This has no effect unless Noise Amplitude is positive

frequency: *Default:* 100, *Range:* 0.1 or greater.
The frequency of the noise. Increase for finer grain noise, decrease for coarser noise. This has no effect unless Noise Amplitude is positive.

frequencyRelX: *Default:* 1, *Range:* 0.01 or greater.
The relative horizontal frequency of the noise. Increase to stretch the noise vertically, decrease to stretch it

horizontally. This has no effect unless Noise Amplitude is positive.

octaves: *Integer*, *Default:* 1, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the magnitude of the previous. This has no effect unless Noise Amplitude is positive.

seed: *Default:* 0.23, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

noiseShift: *X & Y*, *Default:* [0 0], *Range:* any.
The horizontal and vertical translation of the noise texture.

jitterFrames: *Integer*, *Default:* 1, *Range:* 0 or greater.
If this is 0, the noise texture will remain the same for every frame processed. If it is 1, a new noise texture is used for each frame. If it is 2, a new noise texture is used for every other frame, and so on.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_MatteOps(Source, 1.06, shrinkGrow, etc...);`

See Also:

Distort
WarpBubble
Diffuse
Sapphire Plug−ins Introduction

MatteOpsComp

Grows, shrinks, or adds noise to the edges of a Matte input, then uses the result to composite the Front over the Back.

Inputs:

Foreground: The clip to use as foreground.

Background: The clip to use as background.

Matte: *Optional.* The matte input clip to process. It is assumed to have anti−aliased but hard edges, because very soft edges might not be affected in a useful way. Only the alpha channel of this input is used (or green/luma if there is no alpha).

Parameters:

shrinkGrow: *Default:* 0, *Range:* any.
Amount to grow the matte edges in approximate pixels, or shrink if negative.

edgeSoftness: *Default:* 1, *Range:* 0.01 or greater.
The resulting softness of the edges.

postBlur: *Default:* 0, *Range:* 0 or greater.
If positive, the result is blurred by this amount. This is an alternative method for softening the edges.

filter: *Radio buttons, Default: Triangle*.
The type of blur filter to use for the shrink or grow process.

Box: uses a rectangular shaped filter.
Triangle: smoother, uses a pyramid shaped filter.
Gauss: smoothest, uses a gaussian shaped filter.

invertMatte: *Toggle−button, Default:* off.
If enabled, the black and white of the output matte are inverted.

noiseAmplitude: *Default:* 0, *Range:* 0 or greater.
The amount of noise texture to add to the edges.

noiseWidth: *Default:* 2, *Range:* 0 or greater.
The width of the area at the matte edges where the noise is included. This has no effect unless Noise Amplitude is positive

frequency: *Default:* 100, *Range:* 0.1 or greater.
The frequency of the noise. Increase for finer grain noise, decrease for coarser noise. This has no effect unless Noise Amplitude is positive.

frequencyRelX: *Default:* 1, *Range:* 0.01 or greater.
The relative horizontal frequency of the noise. Increase to stretch the noise vertically, decrease to stretch it horizontally. This has no effect unless Noise Amplitude is positive.

octaves: *Integer, Default:* 1, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the magnitude of the previous. This has no effect unless Noise Amplitude is positive.

seed: *Default:* 0.23, *Range:* 0 or greater.

Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

noiseShift: *X & Y*, *Default:* [0 0], *Range:* any.
The horizontal and vertical translation of the noise texture.

jitterFrames: *Integer*, *Default:* 1, *Range:* 0 or greater.
If this is 0, the noise texture will remain the same for every frame processed. If it is 1, a new noise texture is used for each frame. If it is 2, a new noise texture is used for every other frame, and so on.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

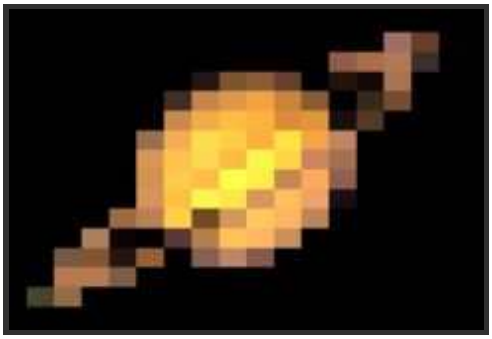
clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_MatteOpsComp(Foreground, Background, Matte, 1.06, shrinkGrow, etc...);`

See Also:

Distort
WarpBubble
Diffuse
Sapphire Plug−ins Introduction

Mosaic



Generates a pixelated version of the source clip. Adjust the size and shape of the blocks using the Pixel Frequency and Pixel Rel Height parameters. Increase the Smooth Colors parameter to cause the colors of nearby pixel blocks to be more consistent, and less flickery over time.

Inputs:

Source: The clip to be processed.

S_Mask: *Optional.* If connected, the effect is applied only at Source areas specified by this Mask input. For gray values in the mask, the pixel blocks are mixed with the original source such that the blocks fade but remain whole. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

pixelFrequency: *Default:* 40, *Range:* 1 or greater.
The frequency of the pixel blocks. Increase for more numerous, smaller pixels.

pixelRelHeight: *Default:* 1/GetDefaultAspect(), *Range:* 0.01 or greater.
The relative height of the pixel blocks. Increase for taller blocks, decrease for wider ones.

pixelShift: *X & Y*, *Default:* [0 0], *Range:* any.
The translation of the pixel pattern.

smoothColors: *Default:* 0, *Range:* 0 or greater.
Blurs the source before pixelating. Increase to cause the colors of nearby pixel blocks to be more consistent, and less flickery over time.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button*, *Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Mosaic(Source, S_Mask, 1.06, pixelFrequency, etc...);`

See Also:

VanGogh
Sketch
HalfTone
HalfToneColor
Etching
ScanLines
FlysEyeHex
JpegDamage
Sapphire Plug−ins Introduction

Psyko: Zebrafy

In the S_Psyko Plugin.

Modulates the brightness of the source clip with a sinusoid to give a black and white solarized look.

Inputs:

Source: The clip to be processed.

Parameters:

sourceBlur: *Default:* .01*width, *Range:* 0 or greater.
Smooths the source edges by this amount.

frequency: *Default:* 4, *Range:* 0.01 or greater.
The frequency of the stripe pattern. Increase for more color cycles.

phaseStart: *Default:* 0, *Range:* any.
The phase shift of the stripe pattern.

phaseSpeed: *Default:* 1, *Range:* any.
The phase speed of the stripe pattern. If non−zero, the stripes are automatically animated to flow at this rate.

color: *Default rgb:* [1 1 1].
Scales the color of the result. For example, if it is yellow [1 1 0], the blue of the result will be 0.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

offset: *Default:* 0, *Range:* any.
Adds this gray value to the result (or subtracts if negative). 0 has no effect, .5 is middle gray, and 1 is white.

scaleBySource: *Default:* 0, *Range:* 0 to 1.
The brightness of the output is scaled down by the original source brightness as this is increased to 1.

scaleBySrcAmp: *Default:* 1, *Range:* 0 or greater.
This amplifies the effect of Scale By Source, so if increased above 1, the middle grays can still retain their full brightness. It has no effect unless Scale By Source is positive.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Psyko(Source, 1.06, "Zebrafy", sourceBlur, etc...);`

See Also:

ZebrafyColor
HalfTone
Etching
ScanLines
Solarize
Sapphire Plug−ins Introduction

Psyko: ZebrafyColor

In the S_Psyko Plugin.

Modulates the brightness of the source clip with sinusoids for each color channel to give a color striped effect.

Inputs:

Source: The clip to be processed.

Parameters:

sourceBlur: *Default:* .01*width, *Range:* 0 or greater.
Smooths the source edges by this amount.

frequency: *Default:* 3, *Range:* 0.01 or greater.
The frequency of the stripe pattern. Increase for more color cycles.

freqRed: *Default:* 1, *Range:* 0 or greater.
The frequency of the red color component.

freqGreen: *Default:* 1.1, *Range:* 0 or greater.
The frequency of the green color component.

freqBlue: *Default:* 1.2, *Range:* 0 or greater.
The frequency of the blue color component.

phaseStart: *Default:* 0, *Range:* any.
The phase shift of the stripe pattern.

phaseRed: *Default:* 0, *Range:* any.
The phase offset of the red color component.

phaseGreen: *Default:* 0, *Range:* any.
The phase offset of the green color component.

phaseBlue: *Default:* 0, *Range:* any.
The phase offset of the blue color component.

phaseSpeed: *Default:* 1, *Range:* any.
The phase speed of the stripe pattern. If non−zero, the stripes are automatically animated to flow at this rate.

color: *Default rgb:* [1 1 1].
Scales the color of the result. For example, if it is yellow [1 1 0], the blue of the result will be 0.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

offset: *Default:* 0, *Range:* any.
Adds this gray value to the result (or subtracts if negative). 0 has no effect, .5 is middle gray, and 1 is white.

saturation: *Default:* 1, *Range:* 0 or greater.
Scales the strength of the colors. Increase for more intense colors, or decrease for muted colors.

scaleBySource: *Default:* 0, *Range:* 0 to 1.
The brightness of the output is scaled down by the original source brightness as this is increased to 1.

scaleBySrcAmp: *Default:* 1, *Range:* 0 or greater.
This amplifies the effect of Scale By Source, so if increased above 1, the middle grays can still retain their full brightness. It has no effect unless Scale By Source is positive.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

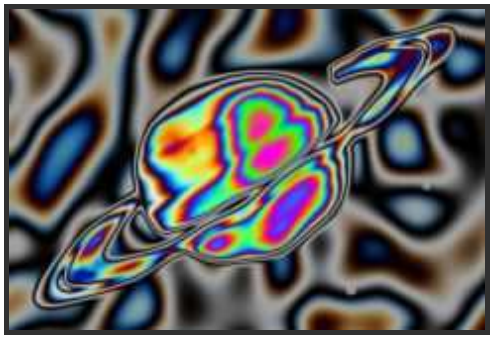
Script Form: `S_Psyko(Source, 1.06, "Zebrafy Color", sourceBlur, etc...);`

See Also:

Zebrafy

PseudoColor
Solarize
PsykoBlobs
PsykoStripes
Sapphire Plug−ins Introduction

PsykoBlobs



In the S_Psyko Plugin.

Combines the source clip with a field of 'blob' shapes and then passes them through a colorization process. The Phase Speed parameter causes the colors to automatically rotate over time.

Inputs:

Source: The clip to be processed.

Parameters:

noiseFreq: *Default:* 4, *Range:* 0.01 or greater.
The spatial frequency of the 'blobs' noise texture. Increase for more blobs, decrease for fewer.

noiseFreqRelx: *Default:* 1, *Range:* 0.01 or greater.
The relative horizontal frequency of the noise texture. Increase to stretch it vertically or decrease to stretch it horizontally.

noiseOctaves: *Integer*, *Default:* 1, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

noiseSeed: *Default:* 0.23, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

noiseShift: *X & Y*, *Default:* [.5*width .5*height], *Range:* any.
Translation offset of the noise texture.

sourceBlur: *Default:* .01*width, *Range:* 0 or greater.
If positive, smooths out the edges of the source by this amount before applying the colorization.

sourceScale: *Default:* 1, *Range:* 0 or greater.
Scales the source values but not the added blobs.

freqColors: *Default:* 4, *Range:* 0 or greater.
The frequency of the color pattern. Increase for a busier texture with more cycles through the spectrum.

freqRed: *Default:* 1, *Range:* 0 or greater.
The frequency of the red color component.

freqGreen: *Default:* 1.1, *Range:* 0 or greater.
The frequency of the green color component.

freqBlue: *Default:* 1.2, *Range:* 0 or greater.
The frequency of the blue color component.

phaseStart: *Default:* 0.5, *Range:* any.
The phase offset of the color patterns.

phaseSpeed: *Default:* 1, *Range:* any.
The phase speed of the color patterns. If non−zero, the phase is automatically animated to give the color pattern a boiling look.

phaseRed: *Default:* 0, *Range:* any.
The phase offset of the red color component.

phaseGreen: *Default:* 0, *Range:* any.
The phase offset of the green color component.

phaseBlue: *Default:* 0, *Range:* any.
The phase offset of the blue color component.

color: *Default rgb:* [1 1 1].
Scales the color of the result. For example, if it is yellow [1 1 0], the blue of the result will be 0.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

offset: *Default:* 0, *Range:* any.
Adds this gray value to the result (or subtracts if negative). 0 has no effect, .5 is middle gray, and 1 is white.

saturation: *Default:* 1, *Range:* 0 or greater.
Scales the strength of the colors. Increase for more intense colors, or decrease for muted colors.

scaleBySource: *Default:* 0, *Range:* 0 to 1.
The brightness of the output is scaled down by the original source brightness as this is increased to 1.

scaleBySrcAmp: *Default:* 1, *Range:* 0 or greater.
This amplifies the effect of Scale By Source, so if increased above 1, the middle grays can still retain their full brightness. It has no effect unless Scale By Source is positive.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

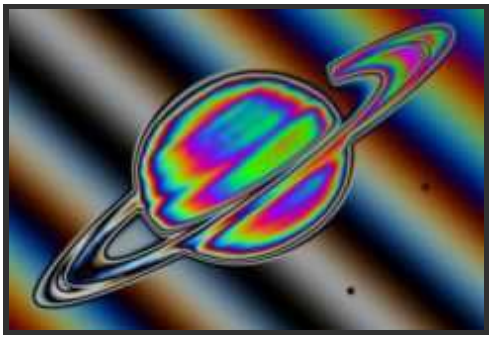
Script Form: `S_Psyko(Source, 1.06, "Psyko Blobs", noiseFreq, etc...);`

See Also:

PsykoStripes

PseudoColor
ZebrafyColor
CloudsPsyko
Sapphire Plug−ins Introduction

PsykoStripes



In the S_Psyko Plugin.

Combines the source clip with a stripe pattern and then passes them through a colorization process. The Phase Speed parameter causes the colors to automatically rotate over time.

Inputs:

Source: The clip to be processed.

Parameters:

stripeDir: *Default:* 45, *Range:* any.
The direction of the stripes, in counter−clockwise degrees from vertical.

stripeMag: *Default:* 0.5, *Range:* 0 or greater.
The magnitude of the stripes. Increase for more cycles of the colors in the stripe direction.

sourceBlur: *Default:* .01*width, *Range:* 0 or greater.
If positive, smooths out the edges of the source by this amount before applying the colorization.

sourceScale: *Default:* 1, *Range:* 0 or greater.
Scales the source values but not the added stripes.

freqColors: *Default:* 3, *Range:* 0 or greater.
The frequency of the color pattern. Increase for a busier texture with more cycles through the spectrum.

freqRed: *Default:* 1, *Range:* 0 or greater.
The frequency of the red color component.

freqGreen: *Default:* 1.1, *Range:* 0 or greater.
The frequency of the green color component.

freqBlue: *Default:* 1.2, *Range:* 0 or greater.
The frequency of the blue color component.

phaseStart: *Default:* 0.5, *Range:* any.
The phase offset of the color patterns.

phaseSpeed: *Default:* 1, *Range:* any.
The phase speed of the color patterns. If non−zero, the phase is automatically animated to give the color pattern a boiling look.

phaseRed: *Default:* 0, *Range:* any.
The phase offset of the red color component.

phaseGreen: *Default:* 0, *Range:* any.
The phase offset of the green color component.

phaseBlue: *Default:* 0, *Range:* any.
The phase offset of the blue color component.

color: *Default rgb:* [1 1 1].
Scales the color of the result. For example, if it is yellow [1 1 0], the blue of the result will be 0.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

offset: *Default:* 0, *Range:* any.
Adds this gray value to the result (or subtracts if negative). 0 has no effect, .5 is middle gray, and 1 is white.

saturation: *Default:* 1, *Range:* 0 or greater.
Scales the strength of the colors. Increase for more intense colors, or decrease for muted colors.

scaleBySource: *Default:* 0, *Range:* 0 to 1.
The brightness of the output is scaled down by the original source brightness as this is increased to 1.

scaleBySrcAmp: *Default:* 1, *Range:* 0 or greater.
This amplifies the effect of Scale By Source, so if increased above 1, the middle grays can still retain their full brightness. It has no effect unless Scale By Source is positive.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Psyko(Source, 1.06, "Psyko Stripes", stripeDir, etc...);`

See Also:

PsykoBlobs
PseudoColor
ZebrafyColor
CloudsPsyko
Sapphire Plug−ins Introduction

RackDefocus

In the S_RackDefocus Plugin.

Generates a defocused version of the source clip using a 'circle of confusion' convolution. This effect is often preferable to a gaussian blur for simulating a real defocused camera lens, because bright spots can be defocused into clean shapes instead of being smoothed away. The Use Gamma parameter can adjust the relative brightness of the blurred highlights.

Inputs:

Source: The clip to be processed.

Parameters:

defocusWidth: *Default:* .01*width, *Range:* 0 or greater.
The width of the defocus. Large values of this parameter are clamped internally if necessary so the width is never greater than the entire size of the image in either dimension.

relHeight: *Default:* 1/GetDefaultAspect(), *Range:* 0.01 or greater.
The relative height of the convolution shape. If it is not 1, circles become ellipses, etc.

gaussBlur: *Default:* 0, *Range:* 0 or greater.
If positive, a gaussian blur is also applied which can smooth out the edges of the shapes. This might also darken the highlights because Gamma is not considered in the gaussian blur.

useGamma: *Default:* 2, *Range:* 0.1 or greater.
Values above 1 cause highlights in the source clip to keep their brightness after the defocus is applied.

shape: *Popup menu, Default: Circle.*
Determines the shape of the simulated camera iris.

Circle: round shape.
Hexagon: six sided shape with a side facing up.
Hexagon R: six sided shape rotated so a vertex faces up.
Octagon: eight sided shape with a side facing up.
Octagon R: eight sided shape rotated so a vertex faces up.

scaleResult: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

offsetDarks: *Default:* 0, *Range:* any.
Adds this gray value to the darker regions of the result. This can be negative to increase contrast.

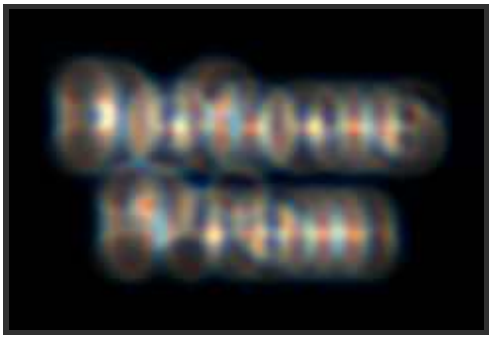
subpixel: *Toggle−button, Default:* off.
Enables defocusing by subpixel amounts. Use this for smoother animation of defocus widths.

Script Form: `S_RackDefocus(Source, 1.06, "Rack Defocus", defocusWidth, etc...);`

See Also:

DefocusPrism
RackDfComp
Blur
BlurChannels

RackDefocus: Prism



In the S_RackDefocus Plugin.

Defocuses the color channels of the source clip into rings of different widths.

Inputs:

Source: The clip to be processed.

Parameters:

defocusWidth: *Default:* .01*width, *Range:* 0 or greater.
The width of the defocus. Large values of this parameter are clamped internally if necessary so the width is never greater than the entire size of the image in either dimension.

relHeight: *Default:* 1/GetDefaultAspect(), *Range:* 0.01 or greater.
The relative height of the convolution shape. If it is not 1, circles become ellipses, etc.

gaussBlur: *Default:* 0, *Range:* 0 or greater.
If positive, a gaussian blur is also applied which can smooth out the edges of the shapes. This might also darken the highlights because Gamma is not considered in the gaussian blur.

useGamma: *Default:* 1, *Range:* 0.1 or greater.
Values above 1 cause highlights in the source clip to keep their brightness after the defocus is applied.

shape: *Popup menu, Default: Circle*.
Determines the shape of the simulated camera iris.

Circle: round shape.
Hexagon: six sided shape with a side facing up.
Hexagon R: six sided shape rotated so a vertex faces up.
Octagon: eight sided shape with a side facing up.
Octagon R: eight sided shape rotated so a vertex faces up.

scaleResult: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

offsetDarks: *Default:* 0, *Range:* any.
Adds this gray value to the darker regions of the result. This can be negative to increase contrast.

subpixel: *Toggle−button, Default:* off.
Enables defocusing by subpixel amounts. Use this for smoother animation of defocus widths.

scaleColor: *Default rgb:* [1 1 1].
Scales the color of the result. For example, if it is yellow [1 1 0], the blue of the result will be 0.

chromaSep: *Default:* 0.3, *Range:* −1 to 1.
The amount of separation between the three color channel rings.

chromaRingthick: *Default:* 0.3, *Range:* 0.01 or greater.
The thickness of each of the three color channel rings.

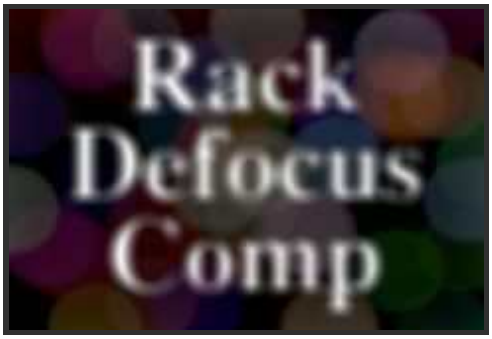
Script Form: `S_RackDefocus(Source, 1.06, "Defocus Prism", defocusWidth, etc...);`

See Also:

RackDefocus

RackDfComp
Blur
BlurChannels
BlurChroma
WarpChroma
Sapphire Plug−ins Introduction

RackDfComp



Composites the Foreground over the Background using a Matte while defocusing both layers by different amounts.

Inputs:

Foreground: The clip to use as foreground.

Background: The clip to use as background.

Matte: *Optional.* Specifies the opacities of the Foreground clip. Only the alpha channel of this input is used (or green/luma if there is no alpha).

Parameters:

defocusForeground: *Default:* .01*width, *Range:* 0 or greater.
The amount to defocus the Foreground and its Matte.

defocusBackground: *Default:* 0, *Range:* 0 or greater.
The amount to defocus the Background.

relHeight: *Default:* 1/GetDefaultAspect(), *Range:* 0.01 or greater.
The relative height of the convolution shape. If it is not 1, circles become ellipses, etc.

useGamma: *Default:* 2, *Range:* 0.1 or greater.
Values above 1 cause highlights in the source clip to keep their brightness after the defocus is applied.

matteGamma: *Default:* 1, *Range:* 0.1 or greater.
The gamma value to use for the defocus of the Matte.

shape: *Popup menu, Default: Circle.*
Determines the shape of the simulated camera iris.

Circle: round shape.
Hexagon: six sided shape with a side facing up.
Hexagon R: six sided shape rotated so a vertex faces up.
Octagon: eight sided shape with a side facing up.
Octagon R: eight sided shape rotated so a vertex faces up.

subpixel: *Toggle−button, Default:* off.
Enables defocusing by subpixel amounts. Use this for smoother animation of defocus widths.

compPremult: *Toggle−button, Default:* on.
Enable this for a better composite if the Foreground pixel values have been pre−multiplied by the Matte pixel values. This is also known as an 'additive' composite.

invertMatte: *Toggle−button, Default:* off.
If enabled, the black and white of the matte are inverted before use.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

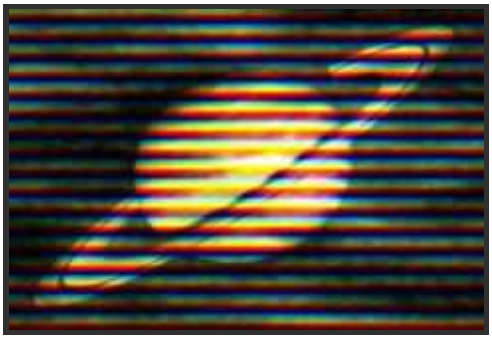
Script Form: `S_RackDfComp(Foreground, Background, Matte, 1.06,`

```
defocusForeground, etc...);
```

See Also:

RackDefocus
DefocusPrism
Blur
Sapphire Plug−ins Introduction

ScanLines



In the S_ScanLines Plugin.

Creates a version of the source clip with scan line pattern resembling a color TV monitor. Increase the Add Noise parameter to also add a grainy effect to the result.

Inputs:

Source: The clip to be processed.

Parameters:

linesFrequency: *Default:* 50, *Range:* 1 or greater.
The frequency of scan lines on the screen. Increase for more lines, decrease for fewer.

linesSharpness: *Default:* 1, *Range:* 0 or greater.
Scales the severity of the lines. Increase for sharper edges, or decrease for a more subtle effect. A sharpness of zero reduces the scan line effect to nothing.

linesAngle: *Default:* 0, *Range:* any.
The angle in degrees of the scan lines. Set to 90 for vertical lines instead of horizontal.

linesShift: *Default:* 0, *Range:* any.
Offsets the position of the pattern of lines. A value of 1.0 shifts one entire scan line over, giving the same result as 0.

shiftRed: *Default:* 0, *Range:* any.
Shifts the red scan lines by this amount, relative to the other lines. Set the red, green, and blue shifts to −.33, .0, and .33 for an out−of−alignment television set look.

shiftGreen: *Default:* 0, *Range:* any.
Shifts the green scan lines by this amount.

shiftBlue: *Default:* 0, *Range:* any.
Shifts the blur scan lines by this amount.

addNoise: *Default:* 0, *Range:* 0 or greater.
If positive, this much color noise is added to the image.

noiseFreqRel: *Default:* 1, *Range:* 0.01 or greater.
The frequency of the noise, relative to the frequency of lines. This has no effect unless the Add Noise parameter above is positive.

scaleColor: *Default rgb:* [1 1 1].
Scales the color of the result. For example, if it is yellow [1 1 0], the blue of the result will be 0.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

offset: *Default:* 0, *Range:* any.
Adds this gray value to the result (or subtracts if negative). 0 has no effect, .5 is middle gray, and 1 is white.

gamma: *Default:* 1.5, *Range:* 0.1 or greater.
Scales the brightness of the image by a curve using this gamma value, allowing adjustment of the middle gray values in the scan lines. This can help make the average brightness of the output match the input.

saturation: *Default:* 1, *Range:* 0 or greater.
Scales the color saturation. Increase for more intense colors. Set to 0 for monochrome.

smoothSource: *Default:* 0, *Range:* 0 or greater.
If positive, the source clip is blurred by this amount before being processed.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

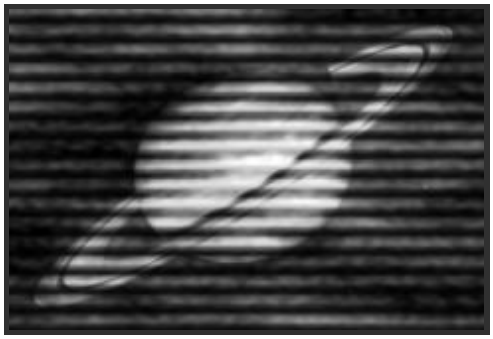
Script Form: `S_ScanLines(Source, 1.06, 0, linesFrequency, etc...);`

See Also:

ScanLinesMono

HalfTone
HalfToneColor
Etching
WipeStripes
JpegDamage
Sapphire Plug−ins Introduction

ScanLinesMono



In the S_ScanLines Plugin.

A monochrome version of ScanLines. Creates a version of the source clip with scan lines pattern resembling a black and white TV monitor. Increase the Add Noise parameter to also add a grainy effect to the result.

Inputs:

Source: The clip to be processed.

Parameters:

linesFrequency: *Default:* 50, *Range:* 1 or greater.
The frequency of scan lines on the screen. Increase for more lines, decrease for fewer.

linesSharpness: *Default:* 1, *Range:* 0 or greater.
Scales the severity of the lines. Increase for sharper edges, or decrease for a more subtle effect. A sharpness of zero reduces the scan line effect to nothing.

linesAngle: *Default:* 0, *Range:* any.
The angle in degrees of the scan lines. Set to 90 for vertical lines instead of horizontal.

linesShift: *Default:* 0, *Range:* any.
Offsets the position of the pattern of lines. A value of 1.0 shifts one entire scan line over, giving the same result as 0.

addNoise: *Default:* 0, *Range:* 0 or greater.
If positive, this much black and white noise is added to the image.

noiseFreqRel: *Default:* 1, *Range:* 0.01 or greater.
The frequency of the noise, relative to the frequency of lines. This has no effect unless the Add Noise parameter above is positive.

color1: *Default rgb:* [1 1 1].
The 'brighter' color of the scan lines pattern.

color0: *Default rgb:* [0 0 0].
The 'darker' color of the scan lines pattern.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

offset: *Default:* 0, *Range:* any.
Adds this gray value to the result (or subtracts if negative). 0 has no effect, .5 is middle gray, and 1 is white.

gamma: *Default:* 1.5, *Range:* 0.1 or greater.
Scales the brightness of the image by a curve using this gamma value, allowing adjustment of the middle gray values in the scan lines. This can help make the average brightness of the output match the input.

smoothSource: *Default:* 0, *Range:* 0 or greater.
If positive, the source clip is blurred by this amount before being processed.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_ScanLines(Source, 1.06, 1, linesFrequency, etc...);`

See Also:

ScanLines

HalfTone
HalfToneColor
Etching
WipeStripes
JpegDamage
Sapphire Plug−ins Introduction

Shake

Applies a shaking motion to the source clip over time with translation, zooming, and/or rotation. The shaking is random but repeatable, so with the same parameters the same shaking motion is generated each time. Turn on Motion Blur and adjust the Mo Blur Length for different amounts of blur. Adjust the Amplitude and Frequency for different shaking speeds and amounts. The Rand parameters give detailed control of the random non−periodic shaking, and the Wave parameters adjust the regular periodic shaking. The X, Y, Z, and Tilt parameters control the horizontal, vertical, zoom, and rotation amounts of shaking respectively.

Inputs:

Source: The clip to shake.

Parameters:

amplitude: *Default:* 1, *Range:* 0 or greater.
Scales the amplitude of the shaking motion.

frequency: *Default:* 8, *Range:* 0 or greater.
Increase for faster shaking, decrease for slower shaking. (Be careful if you animate frequency values because the resulting shake frequency is also affected by the rate of change of the value.)

phase: *Default:* 0, *Range:* any.
Time shift of the shaking motions. (If you animate this value, its rate of change will also affect the apparent frequency.)

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the image. Values greater than 1.0 move it farther away and make it smaller. Values less then 1.0 move the image closer and enlarge it. Zooming in slightly can sometimes be used to hide edge artifacts.

motionBlur: *Toggle−button, Default:* off.
Options for motion blur of the shaking motion.

motionBlurLength: *Default:* 1, *Range:* 0 or greater.
Scales the amount of motion blur. Use around .5 when processing on fields or 1.0 for frames to give realistic motion blur. This parameter has no effect if Motion Blur is *No*.

seed: *Default:* 0, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

xRandAmp: *Default:* .1*width, *Range:* 0 or greater.
Amplitude of horizontal random shaking.

xRandFreq: *Default:* 1, *Range:* 0 or greater.
Frequency of horizontal random shaking.

xWaveAmp: *Default:* 0, *Range:* 0 or greater.
Amplitude of horizontal regular wave shaking.

xWaveFreq: *Default:* 0.5, *Range:* 0 or greater.
Frequency of horizontal regular wave shaking, in cycles per second.

xPhase: *Default:* 0, *Range:* any.
Time shift of the horizontal shaking.

yRandAmp: *Default:* .05*width, *Range:* 0 or greater.
Amplitude of the vertical random shaking.

yRandFreq: *Default:* 1, *Range:* 0 or greater.
Frequency of the vertical random shaking.

yWaveAmp: *Default:* 0, *Range:* 0 or greater.
Amplitude of the vertical regular wave shaking.

yWaveFreq: *Default:* 0.5, *Range:* 0 or greater.
Frequency of the vertical regular wave shaking, in cycles per second.

yPhase: *Default:* 0, *Range:* any.
Time shift of the vertical shaking.

zRandAmp: *Default:* 0, *Range:* 0 or greater.
Amplitude of the zoom random shaking.

zRandFreq: *Default:* 1, *Range:* 0 or greater.
Frequency of the zoom random shaking.

zWaveAmp: *Default:* 0, *Range:* 0 or greater.
Amplitude of the zoom regular wave shaking.

zWaveFreq: *Default:* 0.5, *Range:* 0 or greater.
Frequency of the zoom regular wave shaking, in cycles per second.

zPhase: *Default:* 0, *Range:* any.
Time shift of the zoom shaking.

tiltRandAmp: *Default:* 5, *Range:* 0 or greater.
Amplitude of the rotational random shaking, in degrees.

tiltRandFreq: *Default:* 1, *Range:* 0 or greater.
Frequency of the rotational random shaking.

tiltWaveAmp: *Default:* 0, *Range:* 0 or greater.
Amplitude of the rotational regular wave shaking, in degrees.

tiltWaveFreq: *Default:* 0.5, *Range:* 0 or greater.
Frequency of the rotational regular wave shaking, in cycles per second.

tiltPhase: *Default:* 0, *Range:* any.
Time shift of the rotational shaking.

Script Form: `S_Shake(Source, 1.06, amplitude, etc...);`

See Also:

Flicker
WarpTransform
BlurMotion
BlurMoCurves
Sapphire Plug−ins Introduction

Sharpen

Amplifies the high frequencies in the source clip such as edges and details. Increase the Sharpen Width parameter to sharpen more of the mid range frequencies, and adjust Sharpen Amp to control the amount of sharpening applied.

Inputs:

Source: The clip to be processed.

Parameters:

sharpenAmp: *Default:* 1, *Range:* any.
The amount of sharpening to apply.

sharpenWidth: *Default:* 10, *Range:* 0 or greater.
The width in pixels to perform the sharpen. Increase to sharpen softer edges, decrease to sharpen only the sharper edges.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Sharpen(Source, 1.06, sharpenAmp, etc...);`

See Also:

EdgeDetect
BandPass
Threshold
Sapphire Plug−ins Introduction

SoftFocus

Combines a blurred version of the source with the original to give a 'soft focus' effect. Adjust the Width and Mix parameters to give different types of looks.

Inputs:

Source: The clip to be processed.

Parameters:

softWidth: *Default:* .025*width, *Range:* 0 or greater.
Scales the width of the soft focus blur.

widthRel: *X & Y, Default:* [1 1/GetDefaultAspect()], *Range:* 0 or greater.
The relative horizontal and vertical blur widths. Set Width Rel X to 0 for a vertical−only blur, or set Width Rel Y to 0 for a horizontal−only blur.

mixWithBlurred: *Default:* 0, *Range:* 0 to 1.
If positive, mixes in more of the blurred version of the source.

mixWithSource: *Default:* 0, *Range:* 0 to 1.
If positive, increases the amount of original source in the result.

scaleResult: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

offsetDarks: *Default:* 0, *Range:* any.
Adds this gray value to the darker regions of the result. This can be negative to increase contrast.

subpixel: *Toggle−button, Default:* off.
Enables blurring by subpixel amounts. Use this for smoother animation of the Width parameters.

Script Form: `S_SoftFocus(Source, 1.06, softWidth, etc...);`

See Also:

Blur
Glow
Sapphire Plug−ins Introduction

Sparkles

In the S_Sparkles Plugin.

Generates a field of sparkling glint effects. Adjust the Frequency, Density, and Size parameters for different types of sparkling patterns.

Inputs:

Background: The clip to combine the sparkles with.

S_Mask: *Optional.* If connected, the sparkle colors are scaled by this input. A monochrome Mask can be used to choose the areas that will generate sparkles. A color Mask can be used to selectively adjust the sparkle colors in different regions. The Mask is applied before the sparkles are generated so it will not clip the resulting glint rays.

Parameters:

frequency: *Default:* 25, *Range:* 0.01 or greater.
The frequency of the sparkles. Increase to zoom out, decrease to zoom in.

density: *Default:* 0.65, *Range:* 0 to 1.
Increase to add more sparkles.

seed: *Default:* 0.23, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

size: *Default:* .125*width, *Range:* 0 or greater.
Scales the length of all the glint rays. This and all the size parameters can be adjusted using the Size Widget.

sizeX: *Default:* 1, *Range:* 0 or greater.
Scales the length of the horizontal glint rays.

sizeY: *Default:* 1, *Range:* 0 or greater.
Scales the length of the vertical glint rays.

sizeDiag1: *Default:* 0.5, *Range:* 0 or greater.
Scales the length of the diagonal rays from top right to bottom left.

sizeDiag2: *Default:* 0.5, *Range:* 0 or greater.
Scales the length of the diagonal rays from top left to bottom right.

sizeRed: *Default:* 0.6, *Range:* 0 or greater.
Scales the length of the red component of the rays. If the red, green, and blue sizes are equal the sparkles will be monochrome.

sizeGreen: *Default:* 0.8, *Range:* 0 or greater.
Scales the length of the green component of the rays.

sizeBlue: *Default:* 1, *Range:* 0 or greater.
Scales the length of the blue component of the rays.

color: *Default rgb:* [1 1 1].
Scales the color of all the sparkles.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of all the sparkles.

brightnessX: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the horizontal glint rays.

brightnessY: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the vertical glint rays.

brightnessDiag1: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the diagonal rays from top right to bottom left.

brightnessDiag2: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the diagonal rays from top left to bottom right.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the sparkles. The maximum of the red, green, and blue sparkle brightness is scaled by this value and combined with the background Alpha at each pixel.

shiftStart: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translation offset of the result.

shiftSpeed: *X & Y, Default:* [0 0], *Range:* any.
Translation speed of the result. If non−zero, the result is automatically animated to shift at this rate. The result of animated Speed values may not be intuitive, so for variable speed motion it is usually best to set this to 0 and animate the Shift Start values instead.

sparkleSpeed: *X & Y, Default:* [.05*width 0], *Range:* any.
If non−zero, the sparkles automatically twinkle on and off at this rate.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background before combining with the Sparkles. If 0, the result will contain only the sparkles image over black.

smoothAnim: *Toggle−button, Default:* off.
Enable for more steady animation, especially at high values of Frequency.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Sparkles(Background, S_Mask, 1.06, "Sparkles", frequency, etc...);`

See Also:

SparklesColor Glint
Sapphire Plug−ins Introduction

SparklesColor

In the S_Sparkles Plugin.

Generates a field of sparkling Glint effects with varying colors. Adjust the Frequency, Density, and Size parameters for different types of sparkling patterns.

Inputs:

Background: The clip to combine the sparkles with.

S_Mask: *Optional.* If connected, the sparkle colors are scaled by this input. A monochrome Mask can be used to choose the areas that will generate sparkles. A color Mask can be used to selectively adjust the sparkle colors in different regions. The Mask is applied before the sparkles are generated so it will not clip the resulting glint rays.

Parameters:

frequency: *Default:* 25, *Range:* 0.01 or greater.
The frequency of the sparkles. Increase to zoom out, decrease to zoom in.

density: *Default:* 0.65, *Range:* 0 to 1.
Increase to add more sparkles.

seed: *Default:* 0.23, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

size: *Default:* .125*width, *Range:* 0 or greater.
Scales the length of all the glint rays. This and all the size parameters can be adjusted using the Size Widget.

sizeX: *Default:* 1, *Range:* 0 or greater.
Scales the length of the horizontal glint rays.

sizeY: *Default:* 1, *Range:* 0 or greater.
Scales the length of the vertical glint rays.

sizeDiag1: *Default:* 0.5, *Range:* 0 or greater.
Scales the length of the diagonal rays from top right to bottom left.

sizeDiag2: *Default:* 0.5, *Range:* 0 or greater.
Scales the length of the diagonal rays from top left to bottom right.

sizeRed: *Default:* 0.6, *Range:* 0 or greater.
Scales the length of the red component of the rays. If the red, green, and blue sizes are equal the sparkles will be monochrome.

sizeGreen: *Default:* 0.8, *Range:* 0 or greater.
Scales the length of the green component of the rays.

sizeBlue: *Default:* 1, *Range:* 0 or greater.
Scales the length of the blue component of the rays.

color: *Default rgb:* [1 1 1].

Scales the color of all the sparkles.

colorVariation: *Default:* 1, *Range:* 0 or greater.
Scales the saturation of the sparkles. Increase for more intense colors, decrease for more subtle colors.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of all the sparkles.

brightnessX: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the horizontal glint rays.

brightnessY: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the vertical glint rays.

brightnessDiag1: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the diagonal rays from top right to bottom left.

brightnessDiag2: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the diagonal rays from top left to bottom right.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the sparkles. The maximum of the red, green, and blue sparkle brightness is scaled by this value and combined with the background Alpha at each pixel.

shiftStart: *X & Y*, *Default:* [.5*width .5*height], *Range:* any.
Translation offset of the result.

shiftSpeed: *X & Y*, *Default:* [0 0], *Range:* any.
Translation speed of the result. If non−zero, the result is automatically animated to shift at this rate. The result of animated Speed values may not be intuitive, so for variable speed motion it is usually best to set this to 0 and animate the Shift Start values instead.

sparkleSpeed: *X & Y*, *Default:* [.05*width 0], *Range:* any.
If non−zero, the sparkles automatically twinkle on and off at this rate.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background before combining with the Sparkles. If 0, the result will contain only the sparkles image over black.

smoothAnim: *Toggle−button*, *Default:* off.
Enable for more steady animation, especially at high values of Frequency.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

invertMask: *Toggle−button*, *Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Sparkles(Background, S_Mask, 1.06, "Sparkles Color", frequency, etc...);`

See Also:

Sparkles Glint
Sapphire Plug−ins Introduction

SpotLight

Lights the input clip using one or two spotlights. For each enabled light, the intersection of a 3D light cone with the image plane is calculated using the given light source position, aim location, and beam angle. Ambient light can also be applied to affect the entire source image evenly. A wide variety of lighting shapes can be created by adjusting the parameters provided.

Inputs:

Background: The clip to combine the light with.

Parameters:

light1Enable: *Toggle−button, Default:* on.
Turns on or off this spotlight.

light1Color: *Default rgb:* [1 1 1].
Determines the color of this spotlight.

light1Bright: *Default:* 0.8, *Range:* any.
Scales the brightness of this spotlight. This value can be made negative for a 'dark' spotlight effect.

spreadAngle1: *Default:* 45, *Range:* 0 to 360.
The spread angle of this spotlight beam in degrees. Larger values open up the beam for a larger spot.

softness1: *Default:* 0.3, *Range:* 0.01 to 1.
Determines the amount of penumbra or the softness of the spotlight edges, relative to the Spread Angle. Lower values make crisp edged shapes, higher values make softer shapes.

falloffPower1: *Default:* 0, *Range:* 0 or greater.
Determines how much the spotlight brightness fades with distance. A value of 0 causes no fading, 1 fades the light as distance increases, and 2 fades it faster with distance. A value of 2 is correct for a physically realistic point light.

light1: *X & Y, Default:* [.25*width .75*height], *Range:* any.
The position of this light source relative to the image plane.

light1Z: *Default:* .25*width, *Range:* 10 or greater.
The distance of this light source from the image plane. Decreasing this brings the light source closer to the surface and causes the direction of the beam to be more parallel to the surface, which can stretch the spot into an ellipse or hyperbola shape.

aim1: *X & Y, Default:* [.4*width .5*height], *Range:* any.
This spotlight is directed at this location on the image plane. If this is directly under the Light Source a circular spot will result. When moved away from the Light Source it can also cause the spot to change to an ellipse or hyperbola shape.

light2Enable: *Toggle−button, Default:* off.
Turns on or off the second spotlight.

The remainder of the Light2 parameters are the same as those described above for Light1, but control the second spotlight instead.

ambientBright: *Default:* 0.2, *Range:* any.
The amount of ambient light included in the entire frame. This allows parts of the Background outside of the spotlights to still be visible if desired.

ambientColor: *Default rgb:* [1 1 1].
Determines the color of the ambient light.

allLightsBright: *Default:* 1, *Range:* any.
Scales the brightness of all the spotlights together.

allLightsColor: *Default rgb:* [1 1 1].
Scales the color of all the spotlights together.

allAimsShift: *X & Y, Default:* [0 0], *Range:* any.
Adds this amount to all lights Aim parameters. This can be used to easily make all lights aim at the same location.

allShift: *X & Y, Default:* [0 0], *Range:* any.
Shifts the entire spotlight pattern without changing their shapes by adding this amount to all light and aim positions.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

combine: *Popup menu, Default: Mult.*
Determines how the light is combined with the Background.

> ***Lights Only:*** gives only the light image with no Background.
> ***Mult:*** the light is multiplied by the Background. This is the effect that a real light would typically have.
> ***Add:*** the light is added to the Background.
> ***Screen:*** the light is blended with the Background using a screen operation.
> ***Overlay:*** the light is combined with the Background using an overlay function.

Script Form: `S_SpotLight(Background, 1.06, light1Enable, etc...);`

See Also:

LensFlare
Emboss
Sapphire Plug−ins Introduction

Streaks

Motion blurs the bright areas of the source into streaks between the From and To transformations. This can be used to create an extended film exposure effect, or simulate soft beams of light. From and To parameters do not refer to time. They describe the two transformations in space that determine the style of blur applied to each frame.

Inputs:

Source: The clip to be processed.

S_Mask: *Optional.* If connected, the Source is scaled by the values of this input clip before the areas that get streaked are determined. This can be used to selectively remove or reduce the streaks applied to specific areas of the Source.

Parameters:

streaksBrightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the streaks.

threshold: *Default:* 0.5, *Range:* 0 to 1.
Streaks are generated from locations in the source clip that are brighter than this value. A value of 0.9 causes streaks at only the brightest spots. A value of 0 causes streaks for every non−black area.

thresholdAddColor: *Default rgb:* [0 0 0].
This can be used to raise the threshold on a specific color and thereby reduce the streaks generated on areas of the source clip containing that color.

mixSourceDarks: *Default:* 1, *Range:* 0 to 1.
The dark non−streaked components of the Source are scaled by this amount and added to the result. This allows combining the streaked and non−streaked versions of the source clip.

mixSourceBrights: *Default:* 0, *Range:* 0 to 1.
The original bright components of the Source that were used to generate the streaks are scaled by this amount and added to the result. This allows combining some non−streaked bright areas of the source clip with the output.

scaleResult: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

exposureBias: *Default:* 0, *Range:* 0 to 1.
Determines the variable amount of exposure along the path between the From and To transformations. A value of 0 causes more exposure at the From end, 0.5 causes equal exposure along the path, and 1.0 causes more exposure at the To end. If you have bright spots on a dark background, a 0 value would cause the processed spots to be brighter at the From end and dark at the To end, and a 1.0 value would cause the opposite.

fromZDist: *Default:* 1, *Range:* 0.001 or greater.
The 'distance' of the From transformation. This zooms about the Center location when Shift is 0. Increase to zoom out, decrease to zoom in.

fromRotate: *Default:* 0, *Range:* any.
The rotation angle of the From transformation, in degrees, about the center.

fromShift: *X & Y, Default:* [0 0], *Range:* any.
The horizontal and vertical translations of the From transformation. This can be used for directional motion. If it is non−zero the center location becomes less meaningful.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The center of rotation and zooming, in screen coordinates relative to the center of the frame. The shift values should be zero for this location to make sense.

toZDist: *Default:* 0.8, *Range:* 0.001 or greater.
The 'distance' of the To transformation. Increase to zoom out, or decrease to zoom in.

toRotate: *Default:* 0, *Range:* any.
The rotation angle of the To transformation, in degrees, about the center. Note that if the From and To Rotate angles are very different, the interpolation between them will become less accurate.

toShift: *X & Y, Default:* [0 0], *Range:* any.
The horizontal and vertical translations of the To transformation. This can be used for directional motion. If it is non−zero the center location becomes less meaningful.

combine: *Radio buttons, Default: Add.*
Determines how the streaks are combined with the background.

> ***Add:*** causes the streaks to be added to the background.
> ***Screen:*** performs a blend function which can help prevent overly bright results.

wrap: *Radio buttons, Default: No.*
Determines the method for accessing outside the borders of the source image.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

streaksRes: *Radio buttons, Default: Full.*
Selects the resolution factor for the streaks. This is similar to the general 'Res' factor parameter, but it only affects the streaks: the original mixed with the streaks remains at full resolution. Higher resolutions give better quality, lower resolutions give faster processing.

> ***Full:*** Full resolution is used.
> ***1/2:*** The streaks are calculated at half resolution.
> ***1/4:*** The streaks are calculated at quarter resolution.

subpixel: *Toggle−button, Default:* on.
If enabled, uses a better quality but slightly slower method for rendering the streaks.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

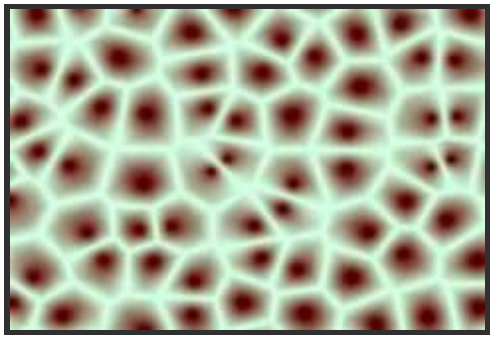
invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Streaks(Source, S_Mask, 1.06, streaksBrightness, etc...);`

See Also:

Rays
EdgeRays
BlurMotion
WarpRepeat
WarpChroma
Sapphire Plug−ins Introduction

TextureCells



In the S_Textures Plugin.

Generates an image of procedural cellular shapes. The Rotate Speed parameter causes the cell centers to rotate within each cell over time.

Inputs:

Background: The clip to combine the texture image with. This may be ignored if the Combine option is set to Texture Only, but the output clip will default to the same length as this clip.

Parameters:

frequency: *Default:* 16, *Range:* 0.01 or greater.
The spatial frequency of the texture. Increase to zoom out, decrease to zoom in.

frequencyRelX: *Default:* 1, *Range:* 0.01 or greater.
The relative horizontal frequency of the texture. Increase to stretch it vertically or decrease to stretch it horizontally.

seed: *Default:* 0.234, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

rotateCenters: *Default:* 0, *Range:* any.
Rotation offset of the cell centers, in counter−clockwise degrees.

rotateSpeed: *Default:* 90, *Range:* any.
The speed of cell center rotation, in counter−clockwise degrees per second. If non−zero, the cell centers are automatically animated to wiggle at this rate.

shift_: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translation offset of the texture. Since the texture is procedurally generated it can be shifted with no repeating units or seams occurring.

brightness1: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of Color1. Increase for more contrast.

color1: *Default rgb:* [1 1 1].
The color of the 'brighter' parts of the texture. The colors of the result are determined by an interpolation between Color0 and Color1.

color0: *Default rgb:* [0 0 0].
The color of the 'darker' parts of the texture.

offset0: *Default:* 0, *Range:* any.
Adds this value to color0. Decrease to a negative value for more contrast.

invert: *Toggle−button, Default:* off.
If enabled, the resulting texture colors are inverted. This is similar to swapping Color0 and Color1.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
The background brightness is scaled by this value before being combined with the texture.

combine: *Popup menu, Default: Screen.*
Determines how the texture is combined with the Background.

Texture Only: gives only the texture image with no Background.
Mult: the texture is multiplied by the Background.
Add: the texture is added to the Background.
Screen: the texture is blended with the Background using a screen operation.
Difference: the result is the difference between the texture and Background.
Overlay: the texture is combined with the Background using an overlay function.

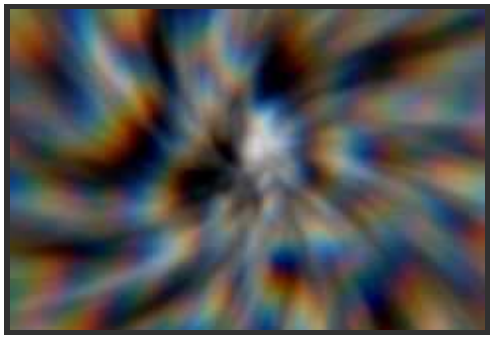
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Textures(Background, 1.06, "Texture Cells", frequency, etc...);`

See Also:

TextureFolded
TextureWeave
TexturePlasma
TextureNoiseEmboss
TextureNoisePaint
TextureSpots
TextureChromaSpiral
TextureMoire
WipeCells
Sapphire Plug−ins Introduction

TextureChromaSpiral



In the S_Textures Plugin.

Creates an abstract texture by applying a WarpChroma effect to a procedurally generated noise texture.

Inputs:

Background: The clip to combine the texture image with. This may be ignored if the Combine option is set to Texture Only, but the output clip will default to the same length as this clip.

Parameters:

noiseFrequency: *Default:* 6, *Range:* 0.01 or greater.
The spatial frequency of the initial noise texture. Increase to zoom out, decrease to zoom in.

noiseOctaves: *Integer, Default:* 3, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

noiseShift: *X & Y, Default:* [0 0], *Range:* any.
Translation offset of the initial noise texture.

noiseSeed: *Default:* 0.23, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The center location of the chroma warp, in screen coordinates relative to the center of the frame.

zDist: *Default:* 0.7, *Range:* 0.001 or greater.
The distance that the chroma warp effect is applied over.

rotate: *Default:* −8, *Range:* any.
The rotation of the spiral, in degrees. Set to 0 for a straight zoom.

steps: *Integer, Default:* 12, *Range:* 3 to 100.
The number of color samples along the spectrum to include. More steps give a smoother result, but require more time to process.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

color: *Default rgb:* [1 1 1].
Scales the color of the result. For example, if it is yellow [1 1 0], the blue of the result will be 0.

offset: *Default:* 0, *Range:* any.
Adds this gray value to the result (or subtracts if negative). 0 has no effect, .5 is middle gray, and 1 is white.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
The background brightness is scaled by this value before being combined with the texture.

combine: *Popup menu, Default: Screen.*
Determines how the texture is combined with the Background.

Texture Only: gives only the texture image with no Background.
Mult: the texture is multiplied by the Background.
Add: the texture is added to the Background.
Screen: the texture is blended with the Background using a screen operation.
Difference: the result is the difference between the texture and Background.
Overlay: the texture is combined with the Background using an overlay function.

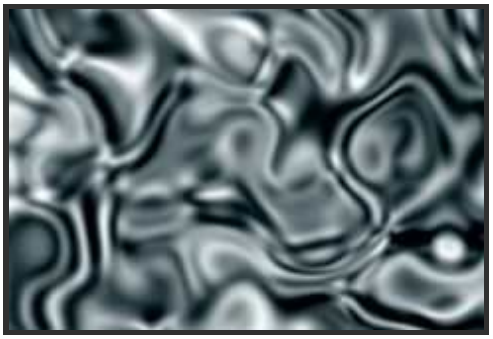
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Textures(Background, 1.06, "Texture Chroma Spiral", noiseFrequency, etc...);`

See Also:

TextureFolded
TextureWeave
TexturePlasma
TextureNoiseEmboss
TextureNoisePaint
TextureSpots
TextureCells
TextureMoire
WarpChroma
Clouds
CloudsVortex
Sapphire Plug−ins Introduction

TextureFolded



In the S_Textures Plugin.

Creates an abstract texture resembling folded cloth or liquid that can be animated to give a dynamic turbulent effect. The Fold Speed parameters cause the pattern to automatically undulate over time.

Inputs:

Background: The clip to combine the texture image with. This may be ignored if the Combine option is set to Texture Only, but the output clip will default to the same length as this clip.

Parameters:

frequency: *Default:* 4, *Range:* 0.01 or greater.
The spatial frequency of the texture. Increase to zoom out, decrease to zoom in.

frequencyRelX: *Default:* 1, *Range:* 0.01 or greater.
The relative horizontal frequency of the texture. Increase to stretch it vertically or decrease to stretch it horizontally.

octaves: *Integer, Default:* 1, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

seed: *Default:* 0.432, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

foldAmp: *Default:* 72, *Range:* any.
The angle of the folding distortions.

foldFreq: *Default:* 0.5, *Range:* 0.01 or greater.
The frequency of the noise used for the folding distortions.

foldOctaves: *Integer, Default:* 1, *Range:* 1 to 10.
The number of octaves of noise to use for the folding distortions.

shift_: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translation offset of the texture. Since the texture is procedurally generated it can be shifted with no repeating units or seams occurring.

foldStart: *X & Y, Default:* [0 0], *Range:* any.
The offset of the folding effect.

foldSpeed: *X & Y, Default:* [.5*width 0], *Range:* any.
The speed of the animated folding effect. If non−zero, the folding effect automatically undulates at this rate.

brightness1: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of Color1. Increase for more contrast.

color1: *Default rgb:* [1 1 1].
The color of the 'brighter' parts of the texture. The colors of the result are determined by an interpolation between Color0 and Color1.

color0: *Default rgb:* [0 0 0].

The color of the 'darker' parts of the texture.

offset0: *Default:* 0, *Range:* any.
Adds this value to color0. Decrease to a negative value for more contrast.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
The background brightness is scaled by this value before being combined with the texture.

combine: *Popup menu, Default: Screen.*
Determines how the texture is combined with the Background.

> ***Texture Only:*** gives only the texture image with no Background.
> ***Mult:*** the texture is multiplied by the Background.
> ***Add:*** the texture is added to the Background.
> ***Screen:*** the texture is blended with the Background using a screen operation.
> ***Difference:*** the result is the difference between the texture and Background.
> ***Overlay:*** the texture is combined with the Background using an overlay function.

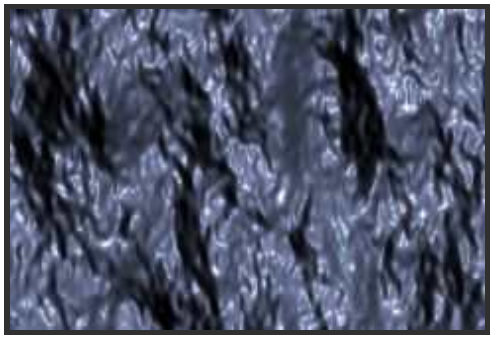
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Textures(Background, 1.06, "Texture Folded", frequency, etc...);`

See Also:

TextureWeave
TexturePlasma
TextureNoiseEmboss
TextureNoisePaint
TextureSpots
TextureCells
TextureChromaSpiral
TextureMoire
Clouds
Sapphire Plug−ins Introduction

TextureNoiseEmboss



In the S_Textures Plugin.

Creates an abstract texture by applying a EmbossShiny effect to a procedurally generated noise texture. Adjust the Light Dir to illuminate the pattern from different angles.

Inputs:

Background: The clip to combine the texture image with. This may be ignored if the Combine option is set to Texture Only, but the output clip will default to the same length as this clip.

Parameters:

frequency: *Default:* 2, *Range:* 0.1 or greater.
The spatial frequency of the texture. Increase to zoom out, decrease to zoom in.

frequencyRelX: *Default:* 1.5, *Range:* 0.01 or greater.
The relative horizontal frequency of the texture. Increase to stretch it vertically or decrease to stretch it horizontally.

octaves: *Integer, Default:* 5, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

seed: *Default:* 0.23, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

bumpsScale: *Default:* 2.5, *Range:* any.
Scales the amplitude of the bump map.

bumpsThreshold: *Default:* 0, *Range:* 0 or greater.
This value is subtracted from the Bumps input before it is used. It can be used to create flat areas resembling 'lakes.'

bumpsSmooth: *Default:* 0, *Range:* 0 or greater.
Smooths the noise texture before applying the Emboss. This can be helpful in removing unwanted artifacts from the noise generation algorithm.

shift_: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translation offset of the texture. Since the texture is procedurally generated it can be shifted with no repeating units or seams occurring.

lightDir: *X & Y, Default:* [.7*width .7*height], *Range:* any.
The direction vector for the light source. Surface shading is calculated using light from this direction shining onto the generated bump map.

hilightBrightness: *Default:* 0.5, *Range:* 0 to 1.
Scales the brightness of the specular highlights.

hilightSize: *Default:* 0.5, *Range:* 0.1 or greater.
Adjusts the size of the specular highlights.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

surfaceColor: *Default rgb:* [0.75 0.75 0.75].
The color of the surface. The final color is affected by both this and the Light Color.

lightColor: *Default rgb:* [1 1 1].
The color of the light source that creates the embossed result.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
The background brightness is scaled by this value before being combined with the texture.

combine: *Popup menu, Default: Screen.*
Determines how the texture is combined with the Background.

> ***Texture Only:*** gives only the texture image with no Background.
> ***Mult:*** the texture is multiplied by the Background.
> ***Add:*** the texture is added to the Background.
> ***Screen:*** the texture is blended with the Background using a screen operation.
> ***Difference:*** the result is the difference between the texture and Background.
> ***Overlay:*** the texture is combined with the Background using an overlay function.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Textures(Background, 1.06, "Texture Noise Emboss", frequency, etc...);`

See Also:

TextureFolded
TextureWeave
TexturePlasma
TextureNoisePaint
TextureSpots
TextureCells
TextureChromaSpiral
TextureMoire
Clouds
EmbossShiny
Sapphire Plug−ins Introduction

TextureNoisePaint

In the S_Textures Plugin.

Creates an abstract texture by applying an AutoPaint effect to a procedurally generated noise texture.

Inputs:

Background: The clip to combine the texture image with. This may be ignored if the Combine option is set to Texture Only, but the output clip will default to the same length as this clip.

Parameters:

noiseFrequency: *Default:* 5, *Range:* 0.1 or greater.
The spatial frequency of the initial noise texture. Increase to zoom out, decrease to zoom in.

noiseFreqRelX: *Default:* 1, *Range:* 0.01 or greater.
The relative horizontal frequency of the initial noise texture. Increase to stretch it vertically or decrease to stretch it horizontally.

noiseOctaves: *Integer*, *Default:* 1, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

noiseSeed: *Default:* 0.23, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

noiseShift: *X & Y*, *Default:* [0 0], *Range:* any.
Translation offset of the initial noise texture.

strokeFrequency: *Default:* 30, *Range:* 0.1 or greater.
The density of brush strokes in the frame. Increase for smaller strokes.

strokeLength: *Default:* −5, *Range:* any.
Determines the length of the brush strokes along the directions of edges in the source clip. If this is negative the strokes will align perpendicular to the edges for a 'HairyPaint' style.

strokeAlign: *Default:* 0, *Range:* 0 or greater.
Increase to smooth out the directions of the strokes so nearby strokes are more parallel.

sharpen: *Default:* 2, *Range:* any.
The amount of post−process sharpening applied.

sharpenWidth: *Default:* 0.1, *Range:* 0 or greater.
The width at which to apply the post−process sharpening filter, relative to the stroke sizes. Higher values affect wider areas from the edges, lower values only affect areas near sharp edges.

jitterFrames: *Integer*, *Default:* 0, *Range:* 0 or greater.
If this is 0, the locations of the strokes will remain the same for every frame processed. If it is 1, the locations of the stokes are re−randomized for each frame. If it is 2, they are re−randomized every second frame, and so on.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

color: *Default rgb:* [1 1 1].
Scales the color of the result. For example, if it is yellow [1 1 0], the blue of the result will be 0.

saturation: *Default:* 0.8, *Range:* 0 or greater.
Scales the color saturation. Increase for more intense colors. Set to 0 for monochrome.

offset: *Default:* 0, *Range:* any.
Adds this gray value to the result (or subtracts if negative). 0 has no effect, .5 is middle gray, and 1 is white.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
The background brightness is scaled by this value before being combined with the texture.

combine: *Popup menu, Default: Screen.*
Determines how the texture is combined with the Background.

> ***Texture Only:*** gives only the texture image with no Background.
> ***Mult:*** the texture is multiplied by the Background.
> ***Add:*** the texture is added to the Background.
> ***Screen:*** the texture is blended with the Background using a screen operation.
> ***Difference:*** the result is the difference between the texture and Background.
> ***Overlay:*** the texture is combined with the Background using an overlay function.

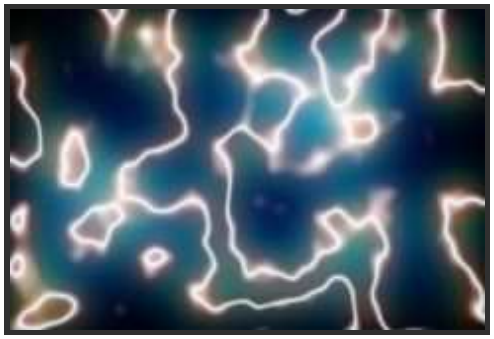
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels. This parameter should not be animated for this effect because the results will pop around the value of 1.5.

Script Form: `S_Textures(Background, 1.06, "Texture Noise Paint", noiseFrequency, etc...);`

See Also:

TextureFolded
TextureWeave
TexturePlasma
TextureNoiseEmboss
TextureSpots
TextureCells
TextureChromaSpiral
TextureMoire
CloudsColorSmooth
VanGogh
Sapphire Plug−ins Introduction

TexturePlasma



In the S_Textures Plugin.

Creates an abstract texture resembling an electrical plasma effect. The Phase Speed parameter causes the pattern to automatically undulate over time.

Inputs:

Background: The clip to combine the texture image with. This may be ignored if the Combine option is set to Texture Only, but the output clip will default to the same length as this clip.

Parameters:

noiseFrequency: *Default:* 1.2, *Range:* 0.01 or greater.
The spatial frequency of the initial noise texture. Increase to zoom out, decrease to zoom in.

noiseFreqRelX: *Default:* 1, *Range:* 0.01 or greater.
The relative horizontal frequency of the initial noise texture. Increase to stretch it vertically or decrease to stretch it horizontally.

noiseOctaves: *Integer, Default:* 4, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

noiseSeed: *Default:* 0.12, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

addGrad: *X & Y, Default:* [0.1 0], *Range:* any.
Determines the amplitude and direction of a gradient which orients the plasma lines. Increasing X makes the lines more vertical, and increasing Y makes them horizontal.

layers: *Default:* 4.5, *Range:* 0 or greater.
The number of layers of plasma lines. Increase for a more striped effect.

threshold: *Default:* 0.5, *Range:* 0 to 1.
Determines the thickness of the plasma lines. Increase for thinner lines, decrease for thicker and brighter ones.

shift_: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translation offset of the texture. Since the texture is procedurally generated it can be shifted with no repeating units or seams occurring.

phaseStart: *Default:* 0, *Range:* any.
Phase offset of the plasma lines.

phaseSpeed: *Default:* 1, *Range:* any.
Phase speed of the plasma lines. If non−zero, the lines are automatically animated to undulate at this rate.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

color: *Default rgb:* [1 1 1].
Scales the color of the result. For example, if it is yellow [1 1 0], the blue of the result will be 0.

glowColor: *Default rgb:* [0.6 0.8 1].
Scales the color of the glow applied to the plasma texture.

glowBrightness: *Default:* 3, *Range:* 0 or greater.
Scales the brightness of the glow applied to the plasma texture.

glowWidth: *Default:* 1, *Range:* 0 or greater.
The width of the glow applied to the plasma texture.

glowWidthRed: *Default:* 0.6, *Range:* 0 or greater.
The relative red width of the glow.

glowWidthGrn: *Default:* 1.2, *Range:* 0 or greater.
The relative green width of the glow.

glowWidthBlue: *Default:* 1.8, *Range:* 0 or greater.
The relative blue width of the glow.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
The background brightness is scaled by this value before being combined with the texture.

combine: *Popup menu, Default: Screen.*
Determines how the texture is combined with the Background.

> ***Texture Only:*** gives only the texture image with no Background.
> ***Mult:*** the texture is multiplied by the Background.
> ***Add:*** the texture is added to the Background.
> ***Screen:*** the texture is blended with the Background using a screen operation.
> ***Difference:*** the result is the difference between the texture and Background.
> ***Overlay:*** the texture is combined with the Background using an overlay function.

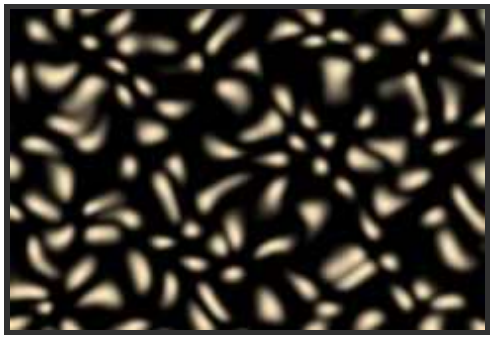
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Textures(Background, 1.06, "Texture Plasma", noiseFrequency, etc...);`

See Also:

TextureFolded
TextureWeave
TextureNoiseEmboss
TextureNoisePaint
TextureSpots
TextureCells
TextureChromaSpiral
TextureMoire
Clouds
Glow
Sapphire Plug−ins Introduction

TextureSpots



In the S_Textures Plugin.

Creates a field of spots that can be distorted and animated. The Warp Speed parameter causes the spots to be distorted over time by a random warping pattern.

Inputs:

Background: The clip to combine the texture image with. This may be ignored if the Combine option is set to Texture Only, but the output clip will default to the same length as this clip.

Parameters:

frequency: *Default:* 8, *Range:* 0.01 or greater.
The spatial frequency of the texture. Increase to zoom out, decrease to zoom in.

radius: *Default:* 1, *Range:* 0 to 2.
The radius of the spots. Adjust this to change the size of the spots without changing the number of spots.

seed: *Default:* 0.23, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

randWarpAmp: *Default:* 0.2, *Range:* 0 or greater.
The amplitude of a bubble warping distortion applied to the spots.

randWarpFreq: *Default:* 1, *Range:* 0.01 or greater.
The spatial frequency of the noise used for the warping distortion. This has no effect unless Rand Warp Amp is positive.

shift_: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translation offset of the texture. Since the texture is procedurally generated it can be shifted with no repeating units or seams occurring.

warpStart: *X & Y, Default:* [0 0], *Range:* any.
The translation offset warping pattern. This has no effect unless Rand Warp Amp is positive.

warpSpeed: *X & Y, Default:* [.25*width 0], *Range:* any.
The translation speed of the warping pattern. If non−zero the spots are animated to wiggle at this rate. This has no effect unless Rand Warp Amp is positive.

brightness1: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of Color1. Increase for more contrast.

color1: *Default rgb:* [1 1 1].
The color of the 'brighter' parts of the texture. The colors of the result are determined by an interpolation between Color0 and Color1.

color0: *Default rgb:* [0 0 0].
The color of the 'darker' parts of the texture.

offset0: *Default:* 0, *Range:* any.
Adds this value to color0. Decrease to a negative value for more contrast.

invert: *Toggle−button, Default:* off.
If enabled, the resulting texture colors are inverted. This is similar to swapping Color0 and Color1.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
The background brightness is scaled by this value before being combined with the texture.

combine: *Popup menu, Default: Screen.*
Determines how the texture is combined with the Background.

> ***Texture Only:*** gives only the texture image with no Background.
> ***Mult:*** the texture is multiplied by the Background.
> ***Add:*** the texture is added to the Background.
> ***Screen:*** the texture is blended with the Background using a screen operation.
> ***Difference:*** the result is the difference between the texture and Background.
> ***Overlay:*** the texture is combined with the Background using an overlay function.

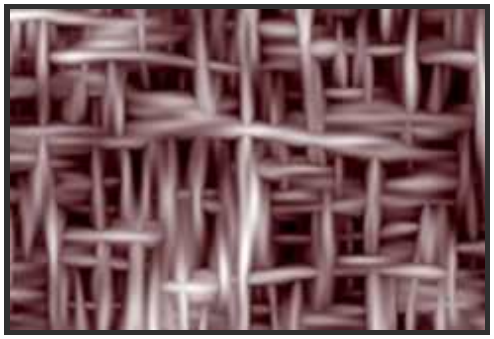
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Textures(Background, 1.06, "Texture Spots", frequency, etc...);`

See Also:

TextureFolded
TextureWeave
TexturePlasma
TextureNoiseEmboss
TextureNoisePaint
TextureCells
TextureChromaSpiral
TextureMoire
WarpBubble
Sapphire Plug−ins Introduction

TextureWeave



In the S_Textures Plugin.

Creates an abstract texture resembling perpendicular woven strands.

Inputs:

Background: The clip to combine the texture image with. This may be ignored if the Combine option is set to Texture Only, but the output clip will default to the same length as this clip.

Parameters:

frequency: *Default:* 20, *Range:* 0.01 or greater.
The spatial frequency of the texture. Increase to zoom out, decrease to zoom in.

hFreqRelX: *Default:* 0.2, *Range:* 0.01 or greater.
The relative horizontal frequency of the horizontal strands. Increase to make shorter, decrease to make longer.

hOctaves: *Integer, Default:* 2, *Range:* 1 to 10.
The number of noise octaves to use for the horizontal strands.

hSpeedX: *Default:* 0, *Range:* any.
The horizontal speed of the horizontal strands. If non−zero, the horizontal strands will automatically crawl along their lengths at this rate.

vFrequency: *Default:* 1, *Range:* 0.01 or greater.
The relative frequency of the vertical strands. Increase to make smaller, decrease to make larger.

vFreqRelY: *Default:* 0.2, *Range:* 0.01 or greater.
The relative vertical frequency of the vertical strands. Increase to make shorter, decrease to make longer.

vOctaves: *Integer, Default:* 2, *Range:* 1 to 10.
The number of noise octaves to use for the vertical strands.

vSpeedY: *Default:* 0, *Range:* any.
The vertical speed of the vertical strands. If non−zero, the vertical strands will automatically crawl along their lengths at this rate.

seed: *Default:* 0.123, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

shift_: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translation offset of the texture. Since the texture is procedurally generated it can be shifted with no repeating units or seams occurring.

shiftSpeed: *X & Y, Default:* [0 0], *Range:* any.
Translation speed of the texture. If non−zero, the result is automatically animated to shift at this rate. The result of animated Speed values may not be intuitive, so for variable speed motion it is usually best to set this to 0 and animate the Shift Start values instead.

sharpen: *Default:* 1, *Range:* any.
The amount of post−process sharpening applied.

sharpenWidth: *Default:* 0.1, *Range:* 0 or greater.

The width at which to apply the post−process sharpening filter, relative to the texture size. Higher values affect wider areas from the edges, lower values only affect areas near sharp edges.

brightness1: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of Color1. Increase for more contrast.

color1: *Default rgb:* [1 1 1].
The color of the 'brighter' parts of the texture. The colors of the result are determined by an interpolation between Color0 and Color1.

color0: *Default rgb:* [0 0 0].
The color of the 'darker' parts of the texture.

offset0: *Default:* 0, *Range:* any.
Adds this value to color0. Decrease to a negative value for more contrast.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
The background brightness is scaled by this value before being combined with the texture.

combine: *Popup menu, Default: Screen.*
Determines how the texture is combined with the Background.

> ***Texture Only:*** gives only the texture image with no Background.
> ***Mult:*** the texture is multiplied by the Background.
> ***Add:*** the texture is added to the Background.
> ***Screen:*** the texture is blended with the Background using a screen operation.
> ***Difference:*** the result is the difference between the texture and Background.
> ***Overlay:*** the texture is combined with the Background using an overlay function.

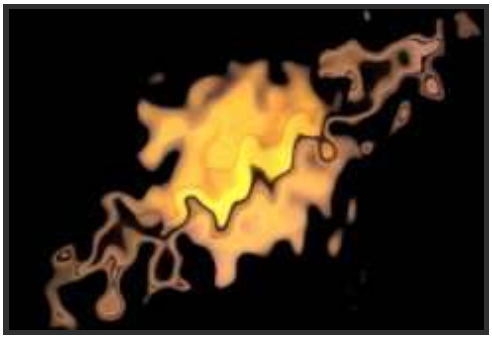
pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Textures(Background, 1.06, "Texture Weave", frequency, etc...);`

See Also:

TextureFolded
TexturePlasma
TextureNoiseEmboss
TextureNoisePaint
TextureSpots
TextureCells
TextureChromaSpiral
TextureMoire
Clouds
Sapphire Plug−ins Introduction

WarpBubble



In the S_Warps Plugin.

Warps the source clip by a smooth noise function. This can be used to create heat diffusion or under water types of effects. The Shift Speed parameters cause the noise pattern to automatically translate over time. Adjust the Amplitude and Frequency parameters to give different types of distortions.

Inputs:

Source: The input clip to be warped.

S_Mask: *Optional.* If connected, the amplitude of warping is scaled by the values of this input clip. Gray values internally scale the warping amplitude rather than simply cross−fading between the effect and the original source to allow more continuous results at the Mask edges and more detailed control over the warping amounts. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

frequency: *Default:* 16, *Range:* 0.01 or greater.
The frequency of the noise pattern. Increase for more and smaller bubbles, decrease for fewer and larger bubbles.

frequencyRelX: *Default:* 1, *Range:* 0.01 or greater.
The relative horizontal frequency of the bubble pattern. Increase for taller bubbles, decrease for wider ones.

amplitude: *Default:* 0.25, *Range:* any.
Scales the amount of warping distortion. Increase for more severe distortion.

octaves: *Integer, Default:* 1, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

seed: *Default:* 0.23, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

rotateWarpDir: *Default:* 0, *Range:* any.
Rotation angle of the warping directions in counter−clockwise degrees.

shiftStart: *X & Y, Default:* [0 0], *Range:* any.
The translation of the bubble pattern.

shiftSpeed: *X & Y, Default:* [.05*width 0], *Range:* any.
If non−zero, the bubble pattern is automatically animated to shift at this speed. The result of animated Speed values may not be intuitive, so for variable speed motion it is usually best to set this to 0 and animate the Shift Start values instead.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the image. Values greater than 1.0 move it farther away and make it smaller. Values less then 1.0 move the image closer and enlarge it. Zooming in slightly can sometimes be used to hide edge artifacts.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].

Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

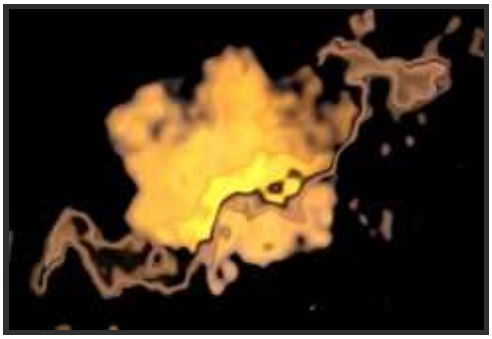
invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Warps(Source, S_Mask, 1.06, "Warp Bubble", frequency, etc...);`

See Also:

WarpTransform
WarpVortex
WarpPerspective
WarpWaves
WarpWaves2
WarpPuddle
WarpBubble2
WarpFishEye
WarpPuff
WarpPolar
WarpRepeat
WarpChroma
DissolveBubble
WipeBubble
Sapphire Plug−ins Introduction

WarpBubble2



In the S_Warps Plugin.

Warps the source clip using two overlapping sets of bubble patterns. This can be used to create heat diffusion or under water types of effects. The Shift Speed parameters cause the noise pattern to automatically translate over time. Adjust the Amplitude and Frequency parameters to give different types of distortions.

Inputs:

Source: The input clip to be warped.

S_Mask: *Optional.* If connected, the amplitude of warping is scaled by the values of this input clip. Gray values internally scale the warping amplitude rather than simply cross−fading between the effect and the original source to allow more continuous results at the Mask edges and more detailed control over the warping amounts. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

aFrequency: *Default:* 4, *Range:* 0.01 or greater.
The frequency of the first set of bubbles.

aAmplitude: *Default:* 0.25, *Range:* any.
The distortion amplitude of the first set of bubbles.

aOctaves: *Integer, Default:* 1, *Range:* 1 to 10.
The number of noise octaves of the first set of bubbles.

aSeed: *Default:* 0.23, *Range:* 0 or greater.
The random number generator seed of the first set of bubbles.

aShiftStart: *X & Y, Default:* [0 0], *Range:* any.
The translation of the first set of bubbles.

aSpeed: *X & Y, Default:* [.05*width 0], *Range:* any.
Automatically animated shift for the first set of bubbles.

bFrequency: *Default:* 4, *Range:* 0.01 or greater.
The frequency of the second set of bubbles.

bAmplitude: *Default:* 0.25, *Range:* any.
The distortion amplitude of the second set of bubbles.

bOctaves: *Integer, Default:* 1, *Range:* 1 to 10.
The number of noise octaves of the second set of bubbles.

bSeed: *Default:* 0.34, *Range:* 0 or greater.
The random number generator seed of the second set of bubbles.

bShiftStart: *X & Y, Default:* [0 0], *Range:* any.
The translation of the second set of bubbles.

bSpeed: *X & Y, Default:* [−.05*width 0], *Range:* any.
Automatically animated shift for the second set of bubbles.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the image. Values greater than 1.0 move it farther away and make it smaller. Values less then 1.0 move the image closer and enlarge it. Zooming in slightly can sometimes be used to hide edge artifacts.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

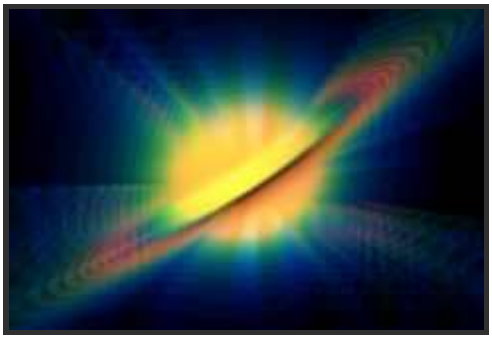
invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Warps(Source, S_Mask, 1.06, "Warp Bubble2", aFrequency, etc...);`

See Also:

WarpTransform
WarpVortex
WarpPerspective
WarpWaves
WarpWaves2
WarpPuddle
WarpBubble
WarpFishEye
WarpPuff
WarpPolar
WarpRepeat
WarpChroma
DissolveBubble
WipeBubble
Sapphire Plug−ins Introduction

WarpChroma



In the S_Warps Plugin.

Separates the source clip into spectral bands and warps them by different amounts. The red is warped by the From transformation, the blue by the To transformation, with the other colors of the spectrum in between. The From and To parameters do not refer to time. They describe the two transformations in space that determine the sequence of warps applied to each color.

Inputs:

Source: The input clip to be warped.

S_Mask: *Optional.* If connected, the amplitude of warping is scaled by the values of this input clip. Gray values internally scale the warping amplitude rather than simply cross−fading between the effect and the original source to allow more continuous results at the Mask edges and more detailed control over the warping amounts. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The center of rotation and zooming, in screen coordinates relative to the center of the frame. The shift values should be zero for this location to make sense.

steps: *Integer, Default:* 8, *Range:* 3 to 100.
The number of spectrum samples to include along the path between the From (red) and To (blue) transformations. More steps give a smoother result, but require more time to process.

fromZDist: *Default:* 1.5, *Range:* 0.001 or greater.
The 'distance' of the From transformation. This zooms about the Center location when Shift is 0. Increase to zoom out, decrease to zoom in.

fromRotate: *Default:* 0, *Range:* any.
The rotation angle of the From transformation, in degrees, about the center.

fromShift: *X & Y, Default:* [0 0], *Range:* any.
The horizontal and vertical translations of the From transformation. This can be used for directional motion. If it is non−zero the center location becomes less meaningful.

toZDist: *Default:* 1, *Range:* 0.001 or greater.
The 'distance' of the To transformation. Increase to zoom out, or decrease to zoom in.

toRotate: *Default:* 0, *Range:* any.
The rotation angle of the To transformation, in degrees, about the center. Note that if the From and To Rotate angles are very different, the interpolation between them will become less accurate.

toShift: *X & Y, Default:* [0 0], *Range:* any.
The horizontal and vertical translations of the To transformation. This can be used for directional motion. If it is non−zero the center location becomes less meaningful.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the result.

color1: *Default rgb:* [1 0 0].
The color at the From transformation.

color2: *Default rgb:* [0 1 0].
The color midway between the From and To transformations.

color3: *Default rgb:* [0 0 1].
The color at the To transformation.

whiteBalance: *Toggle−button, Default:* off.
When enabled, the three colors are adjusted internally so they sum to white. In this case, the colors of unwarped regions are not affected and the average color of the result remains the same.

wrap: *X & Y, Radio buttons, Default:* [*No No*].
Determines the method for accessing outside the borders of the source image.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* off.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Warps(Source, S_Mask, 1.06, "Warp Chroma", centerX, etc...);`

See Also:

WarpTransform
WarpVortex
WarpPerspective
WarpWaves
WarpWaves2
WarpPuddle
WarpBubble
WarpBubble2
WarpFishEye
WarpPuff
WarpPolar
WarpRepeat
DistortChroma
DefocusPrism
BlurMotion
Streaks
EdgeRays
Sapphire Plug−ins Introduction

WarpDrops

Warps the source clip by multiple patterns of concentric waves emanating from multiple center locations. Each area in the Centers input clip brighter than the value of Threshold Cntrs, generates an independent pattern of concentric waves, and the total brightness of each area scales the warping magnitude of those waves. If the Centers image is complex, the number and locations of resulting centers can be fairly sensitive to the threshold value. Try using just solid black with a few white dots for the Centers input. If you only need a single set of waves, you can use the WarpPuddle effect instead.

Inputs:

Source: The input clip to be warped.

Centers: Determines the centers of the wave patterns. Each area in this clip brighter than the value of Threshold Cntrs, generates an independent pattern of concentric waves. The total brightness of the area (brightness x area) scales the warping magnitude of those waves. This clip is often a painted image of dots of different sizes and brightnesses. If the painted centers move over time, the effect centers will move with them.

S_Mask: *Optional.* If connected, the amplitude of warping is scaled by the values of this input clip. Gray values internally scale the warping amplitude rather than simply cross−fading between the effect and the original source to allow more continuous results at the Mask edges and more detailed control over the warping amounts. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

frequency: *Default:* 10, *Range:* 0.01 or greater.
The frequency of the waves. Increase for more waves, decrease for fewer.

amplitude: *Default:* 1, *Range:* any.
Scales the amount of warping distortion. Increase for more severe distortion.

relHeight: *Default:* 1, *Range:* 0.01 or greater.
The relative height of the concentric wave pattern.

rotateRelH: *Default:* 0, *Range:* any.
Rotation in degrees of the wave patterns, about each center. This has no effect if the Rel Height parameter is 1.0.

thresholdCntrs: *Default:* 0.6, *Range:* 0 to 1.
Areas brighter than this value are used as centers for the waves. A center is generated from the centroid of each set of connected pixels above this value.

maxCenters: *Integer, Default:* 20, *Range:* 1 or greater.
The maximum total number of centers to use. This can be used for testing or to avoid overly large numbers of centers.

phaseStart: *Default:* 0, *Range:* any.
The phase shift of the waves.

phaseSpeed: *Default:* 0, *Range:* any.
The speed of the waves. If this is positive the waves automatically travel outwards from the center at this rate.

innerRadius: *Default:* 0, *Range:* any.
The distance from the puddle center where the wave distortion is phased in. No waves are generated inside this radius.

innerSoftness: *Default:* .05*width, *Range:* 1 or greater.
The width of the region at the Inner Radius over which the wave distortion is phased in.

outerRadius: *Default:* .25*width, *Range:* 0 or greater.
The distance from the puddle center where the wave distortion is phased out. No waves are generated outside this radius.

outerSoftness: *Default:* .25*width, *Range:* 2 or greater.
The width of the region at the Outer Radius over which the wave distortion is phased out.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the image. Values greater than 1.0 move it farther away and make it smaller. Values less then 1.0 move the image closer and enlarge it. Zooming in slightly can sometimes be used to hide edge artifacts.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_WarpDrops(Source, Centers, S_Mask, 1.06, frequency, etc...);`

See Also:

WarpPuddle
WarpWaves
Sapphire Plug−ins Introduction

WarpFishEye

In the S_Warps Plugin.

Expands the center of the source clip as if viewed through a fish−eye lens. Adjust the Amount parameter to give more or less distortion. Turn off the Wrap options to give transparency beyond the borders of the input clip instead of reflected copies.

Inputs:

Source: The input clip to be warped.

S_Mask: *Optional.* If connected, the amplitude of warping is scaled by the values of this input clip. Gray values internally scale the warping amplitude rather than simply cross−fading between the effect and the original source to allow more continuous results at the Mask edges and more detailed control over the warping amounts. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

amount: *Default:* 1, *Range:* any.
The amplitude of the fish−eye warping. Try this negative with a large Z Dist for some wacky 'bug eye' distortions.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The center of the fish−eye warping function, in screen coordinates relative to the center of the frame.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the image. Values greater than 1.0 move it farther away and make it smaller. Values less then 1.0 move the image closer and enlarge it. Zooming in slightly can sometimes be used to hide edge artifacts.

rotate: *Default:* 0, *Range:* any.
Rotates the result about the center location by this many counter−clockwise degrees.

shiftOrig: *X & Y, Default:* [0 0], *Range:* any.
Translates the source image before the fish−eye warping is applied.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked

and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Warps(Source, S_Mask, 1.06, "Warp Fish Eye", amount, etc...);`

See Also:

WarpTransform
WarpVortex
WarpPerspective
WarpWaves
WarpWaves2
WarpPuddle
WarpBubble
WarpBubble2
WarpPuff
WarpPolar
WarpRepeat
WarpChroma
Distort
Sapphire Plug−ins Introduction

WarpPerspective

In the S_Warps Plugin.

Transforms the source clip onto a 3D plane with perspective. Adjust the Latitude, Swing, and Roll parameters to rotate the image on various axes, and adjust Shift and Z Dist to translate and zoom. Turn off the Wrap options to give a single non−repeated copy of the image.

Inputs:

Source: The input clip to be warped.

S_Mask: *Optional.* If connected, the amplitude of warping is scaled by the values of this input clip. Gray values internally scale the warping amplitude rather than simply cross−fading between the effect and the original source to allow more continuous results at the Mask edges and more detailed control over the warping amounts. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

latitude: *Default:* 35, *Range:* −89 to 89.
Positive latitude tilts the image down and negative tilts it up. Keep latitude in the range of around −35 to 35 degrees to avoid aliasing towards the horizon.

swing: *Default:* 0, *Range:* any.
Rotation of the image in degrees in its initial frame.

roll: *Default:* 0, *Range:* any.
Tilts the result from side to side, in counter−clockwise degrees.

zDist: *Default:* 3, *Range:* 0.001 or greater.
Scales the 'distance' of the image. Values greater than 1.0 move it farther away and make it smaller. Values less then 1.0 move the image closer and enlarge it.

teleLensWidth: *Default:* 1, *Range:* 0.2 to 3.
The amount of lens telescoping. Increase to zoom in with less perspective, decrease for a wider viewing angle with more perspective.

shiftOrig: *X & Y, Default:* [0 0], *Range:* any.
Translates the image in its initial frame.

wrap: *X & Y, Radio buttons, Default:* [*Tile Tile*].
Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.

The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button*, *Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Warps(Source, S_Mask, 1.06, "Warp Perspective", latitude, etc...);`

See Also:

WarpTransform
WarpVortex
WarpWaves
WarpWaves2
WarpPuddle
WarpBubble
WarpBubble2
WarpFishEye
WarpPuff
WarpPolar
WarpRepeat
WarpChroma
CloudsPerspective
Sapphire Plug−ins Introduction

WarpPolar

In the S_Warps Plugin.

Warps the source clip into a rounded disk shape. The vertical direction of the source image is mapped between the Inner Radius and Outer Radius, and the horizontal direction is rotated about the center based on the number of Angle Repeats and offset by Angle.

Inputs:

Source: The input clip to be warped.

S_Mask: *Optional.* If connected, the amplitude of warping is scaled by the values of this input clip. Gray values internally scale the warping amplitude rather than simply cross−fading between the effect and the original source to allow more continuous results at the Mask edges and more detailed control over the warping amounts. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

angle: *Default:* 0, *Range:* any.
Rotation of the result, in counter−clockwise degrees.

angleRepeats: *Default:* 1, *Range:* 0.01 or greater.
The number of copies of the source image to wrap around. This should be an integer to avoid a seam where the first copy connects to the last.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The center of the disk, in screen coordinates relative to the center of the frame.

stretch: *X & Y, Default:* [1 1], *Range:* 0.01 or greater.
Scales the horizontal or vertical size of the disk shape.

innerRadius: *Default:* .1*width, *Range:* any.
The distance from the center where the bottom edge of the source clip is mapped.

outerRadius: *Default:* .4*width, *Range:* any.
The distance from the center where the top edge of the input clip is mapped.

wrap: *X & Y, Radio buttons, Default:* [*Tile No*].
Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

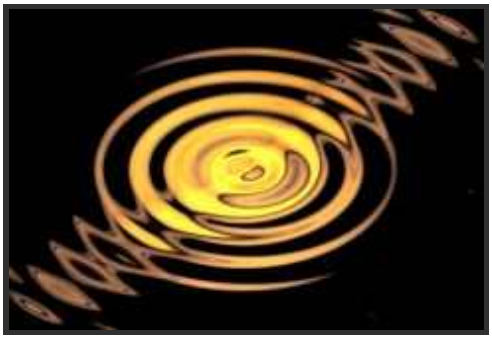
invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Warps(Source, S_Mask, 1.06, "Warp Polar", angle, etc...);`

See Also:

WarpTransform
WarpVortex
WarpPerspective
WarpWaves
WarpWaves2
WarpPuddle
WarpBubble
WarpBubble2
WarpFishEye
WarpPuff
WarpRepeat
WarpChroma
KaleidoPolar
Sapphire Plug−ins Introduction

WarpPuddle



In the S_Warps Plugin.

Warps the source clip by a pattern of concentric waves. The Phase Speed parameter causes the waves to automatically move outwards from the center over time. Adjust the Inner and Outer Radius parameters to limit the area where the waves appear. Increase the Inner and Outer softness for smoother transitions between where the waves appear and don't appear.

Inputs:

Source: The input clip to be warped.

S_Mask: *Optional.* If connected, the amplitude of warping is scaled by the values of this input clip. Gray values internally scale the warping amplitude rather than simply cross−fading between the effect and the original source to allow more continuous results at the Mask edges and more detailed control over the warping amounts. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

frequency: *Default:* 10, *Range:* 0.01 or greater.
The frequency of the waves. Increase for more waves, decrease for fewer.

amplitude: *Default:* 0.1, *Range:* any.
Scales the amount of warping distortion. Increase for more severe distortion.

relHeight: *Default:* 0.75, *Range:* 0.01 or greater.
The relative height of the concentric wave pattern.

rotatePuddle: *Default:* 0, *Range:* any.
Rotates the puddle pattern by this many counter−clockwise degrees after the Rel Height stretching has been applied. This has no effect when Rel Height is 1.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The center of the puddle, in screen coordinates relative to the center of the frame.

phaseStart: *Default:* 0, *Range:* any.
The phase shift of the waves.

phaseSpeed: *Default:* 1, *Range:* any.
The speed of the waves. If this is positive the waves automatically travel outwards from the center at this rate.

innerRadius: *Default:* 0, *Range:* any.
The distance from the puddle center where the wave distortion is phased in. No waves are generated inside this radius.

innerSoftness: *Default:* .05*width, *Range:* 2 or greater.
The width of the region at the Inner Radius over which the wave distortion is phased in.

outerRadius: *Default:* .75*width, *Range:* 0 or greater.
The distance from the puddle center where the wave distortion is phased out. No waves are generated outside this radius.

outerSoftness: *Default:* .2*width, *Range:* 2 or greater.
The width of the region at the Outer Radius over which the wave distortion is phased out.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the image. Values greater than 1.0 move it farther away and make it smaller. Values less then 1.0 move the image closer and enlarge it. Zooming in slightly can sometimes be used to hide edge artifacts.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

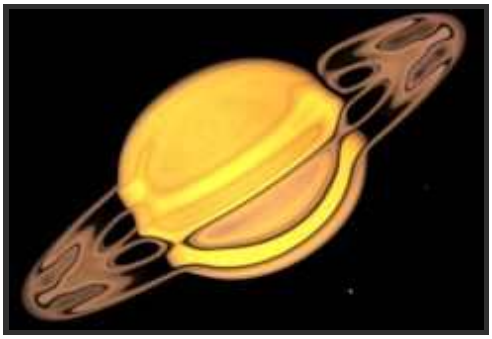
invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Warps(Source, S_Mask, 1.06, "Warp Puddle", frequency, etc...);`

See Also:

WarpTransform
WarpVortex
WarpPerspective
WarpWaves
WarpWaves2
WarpBubble
WarpBubble2
WarpFishEye
WarpPuff
WarpPolar
WarpRepeat
WarpChroma
DissolvePuddle
Sapphire Plug−ins Introduction

WarpPuff



In the S_Warps Plugin.

Warps the source clip based on its gradient. By default, brighter areas are puffed out and darker areas are shrunk. This is similar to applying Distort effect to an image using itself as the lens.

Inputs:

Source: The input clip to be warped.

S_Mask: *Optional.* If connected, the amplitude of warping is scaled by the values of this input clip. Gray values internally scale the warping amplitude rather than simply cross−fading between the effect and the original source to allow more continuous results at the Mask edges and more detailed control over the warping amounts. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

direction: *Radio buttons, Default: Puff.*
Determines which type of areas of the source clip are puffed.

Puff: expands brighter areas and shrinks darker ones.
Shrivel: shrinks brighter areas and expands darker ones.

amount: *Default:* 0.5, *Range:* any.
Scales the amount of distortion. This can also be negative to turn puffs into shrivels and vice versa.

smoothness: *Default:* .05*width, *Range:* 0 or greater.
Blurs the source clip by this amount before determining the warp directions and amounts.

rotateWarpDir: *Default:* 0, *Range:* any.
Rotates the direction of the warping. This can cause areas of similar brightness to be twisted instead of just expanded or shrunk.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Warps(Source, S_Mask, 1.06, "Warp Puff", direction, etc...);`

See Also:

WarpTransform
WarpVortex
WarpPerspective
WarpWaves
WarpWaves2
WarpPuddle
WarpBubble
WarpBubble2
WarpFishEye
WarpPolar
WarpRepeat
WarpChroma
Distort
Sapphire Plug−ins Introduction

WarpRepeat

In the S_Warps Plugin.

Transforms the source input multiple times and averages the results. The From and To parameters do not refer to time. They describe the two transformations in space that determine the sequence of repeated warps applied to each frame.

Inputs:

Source: The input clip to be warped.

S_Mask: *Optional.* If connected, the amplitude of warping is scaled by the values of this input clip. Gray values internally scale the warping amplitude rather than simply cross−fading between the effect and the original source to allow more continuous results at the Mask edges and more detailed control over the warping amounts. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The center of rotation and zooming, in screen coordinates relative to the center of the frame. The shift values should be zero for this location to make sense.

exposureBias: *Default:* 0.5, *Range:* 0 to 1.
Determines the variable amount of exposure along the path between the From and To transformations. A value of 0 causes more exposure at the From end, 0.5 causes equal exposure along the path, and 1.0 causes more exposure at the To end. If you have bright spots on a dark background, a 0 value would cause the processed spots to be brighter at the From end and dark at the To end, and a 1.0 value would cause the opposite.

steps: *Integer, Default:* 3, *Range:* 2 to 100.
The number of times the input image is sampled along the path between the From and To transformations. More steps require more processing time.

fromZDist: *Default:* 1.5, *Range:* 0.001 or greater.
The 'distance' of the From transformation. This zooms about the Center location when Shift is 0. Increase to zoom out, decrease to zoom in.

fromRotate: *Default:* 0, *Range:* any.
The rotation angle of the From transformation, in degrees, about the center.

fromShift: *X & Y, Default:* [0 0], *Range:* any.
The horizontal and vertical translations of the From transformation. This can be used for directional motion. If it is non−zero the center location becomes less meaningful.

toZDist: *Default:* 1, *Range:* 0.001 or greater.
The 'distance' of the To transformation. Increase to zoom out, or decrease to zoom in.

toRotate: *Default:* 0, *Range:* any.
The rotation angle of the To transformation, in degrees, about the center. Note that if the From and To Rotate angles are very different, the interpolation between them will become less accurate.

toShift: *X & Y, Default:* [0 0], *Range:* any.

The horizontal and vertical translations of the To transformation. This can be used for directional motion. If it is non−zero the center location becomes less meaningful.

wrap: *X & Y, Radio buttons, Default:* [*No No*].
Determines the method for accessing outside the borders of the source image.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* off.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Warps(Source, S_Mask, 1.06, "Warp Repeat", centerX, etc...);`

See Also:

WarpTransform
WarpVortex
WarpPerspective
WarpWaves
WarpWaves2
WarpPuddle
WarpBubble
WarpBubble2
WarpFishEye
WarpPuff
WarpPolar
WarpChroma
BlurMotion
Streaks
EdgeRays
Sapphire Plug−ins Introduction

WarpTransform

In the S_Warps Plugin.

Warps the source clip by a combination of linear transformations including scale, shear, zoom, rotation, and translation.

Inputs:

Source: The input clip to be warped.

S_Mask: *Optional.* If connected, the amplitude of warping is scaled by the values of this input clip. Gray values internally scale the warping amplitude rather than simply cross−fading between the effect and the original source to allow more continuous results at the Mask edges and more detailed control over the warping amounts. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

scale: *X & Y, Default:* [1 scaleX], *Range:* any.
Scales the relative horizontal or vertical size of the source image.

shift_: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translates the source image.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the image. Values greater than 1.0 move it farther away and make it smaller. Values less then 1.0 move the image closer and enlarge it. Note that Scale X and Y also scale the size of the image, but in an inverse way and on each axis.

rotate: *Default:* 0, *Range:* any.
Rotates the source image by the specified angle in counter−clockwise degrees.

shear: *X & Y, Default:* [0 0], *Range:* any.
Shears the source image horizontally or vertically.

wrap: *X & Y, Radio buttons, Default:* [*No No*].
Determines the method for accessing outside the borders of the source image.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Warps(Source, S_Mask, 1.06, "Warp Transform", scaleX, etc...);`

See Also:

WarpVortex
WarpPerspective
WarpWaves
WarpWaves2
WarpPuddle
WarpBubble
WarpBubble2
WarpFishEye
WarpPuff
WarpPolar
WarpRepeat
WarpChroma
Shake
Sapphire Plug−ins Introduction

WarpVortex

In the S_Warps Plugin.

Twists the source clip into a vortex, about a given Center location. Use the Vortex Start parameter to adjust the amount of vortexing, and use Angle Offset to also apply a normal rotation. Vortex Speed can be used to automatically animate the amount of vortexing.

Inputs:

Source: The input clip to be warped.

S_Mask: *Optional.* If connected, the amplitude of warping is scaled by the values of this input clip. Gray values internally scale the warping amplitude rather than simply cross−fading between the effect and the original source to allow more continuous results at the Mask edges and more detailed control over the warping amounts. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

vortexStart: *Default:* 36, *Range:* any.
The amount of vortex rotation, in approximate counter−clockwise degrees at the edge of the frame.

vortexSpeed: *Default:* 0, *Range:* any.
The speed of the vortex rotation, in approximate degrees per second at the edge of the frame. If non−zero, the vortexing is automatically animated at this rate.

angleOffset: *Default:* 0, *Range:* any.
If non−zero, a rotation is combined with the vortex. Make negative to rotate the inner and outer regions in opposite directions.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The center of the vortex, in screen coordinates relative to the center of the frame.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the image. Values greater than 1.0 move it farther away and make it smaller. Values less then 1.0 move the image closer and enlarge it.

innerRadius: *Default:* .02*width, *Range:* 0 or greater.
The radius from the center at which the vortexing is phased in. This can be used to reduce excessive distortion and aliasing at the very center of the vortex.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

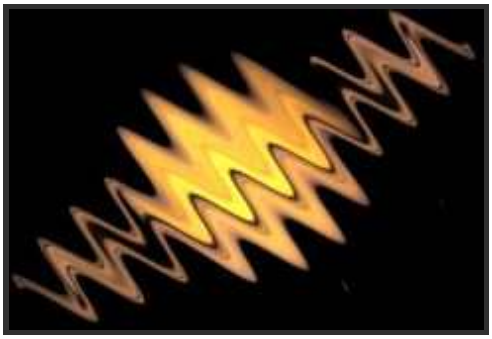
invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Warps(Source, S_Mask, 1.06, "Warp Vortex", vortexStart, etc...);`

See Also:

WarpTransform
WarpPerspective
WarpWaves
WarpWaves2
WarpPuddle
WarpBubble
WarpBubble2
WarpFishEye
WarpPuff
WarpPolar
WarpRepeat
WarpChroma
DissolveVortex
Sapphire Plug−ins Introduction

WarpWaves



In the S_Warps Plugin.

Warps the source clip by a wave pattern. You can make the waves move over time by increasing the Phase Speed parameter, or by animating the value of Phase Start.

Inputs:

Source: The input clip to be warped.

S_Mask: *Optional.* If connected, the amplitude of warping is scaled by the values of this input clip. Gray values internally scale the warping amplitude rather than simply cross−fading between the effect and the original source to allow more continuous results at the Mask edges and more detailed control over the warping amounts. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

frequency: *Default:* 8, *Range:* 0.1 or greater.
The frequency of the waves. Increase for more waves, decrease for fewer.

amplitude: *Default:* 0.1, *Range:* any.
Scales the amount of warping distortion. Increase for more severe distortion.

angle: *Default:* 45, *Range:* any.
The rotation angle of the wave pattern in counter−clockwise degrees. If angle is 0, the waves move to the right and are aligned vertically.

displaceAngle: *Default:* 90, *Range:* any.
The warping direction in degrees relative to the angle of the waves. 0 gives compression−expansion waves, and 90 gives side to side waves.

phaseStart: *Default:* 0, *Range:* any.
The phase shift of the waves. The wave pattern is translated in the direction of Angle by this amount.

phaseSpeed: *Default:* 0, *Range:* any.
The phase speed of the waves. If this is non−zero the wave pattern automatically travels at this rate.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the image. Values greater than 1.0 move it farther away and make it smaller. Values less then 1.0 move the image closer and enlarge it. Zooming in slightly can sometimes be used to hide edge artifacts.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image.

No: gives the background pixel color beyond the borders.
Tile: repeats a copy of the image.
Reflect: repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

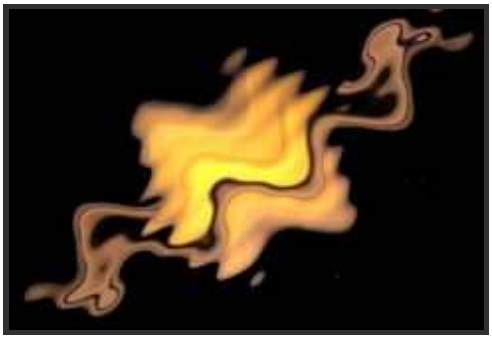
invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Warps(Source, S_Mask, 1.06, "Warp Waves", frequency, etc...);`

See Also:

WarpTransform
WarpVortex
WarpPerspective
WarpWaves2
WarpPuddle
WarpBubble
WarpBubble2
WarpFishEye
WarpPuff
WarpPolar
WarpRepeat
WarpChroma
DissolveWaves
Sapphire Plug−ins Introduction

WarpWaves2



In the S_Warps Plugin.

Warps the source clip using two sets of overlapping wave patterns. You can make the waves move over time by increasing the Phase Speed parameters, or by animating the value of the Phase Start parameters.

Inputs:

Source: The input clip to be warped.

S_Mask: *Optional.* If connected, the amplitude of warping is scaled by the values of this input clip. Gray values internally scale the warping amplitude rather than simply cross−fading between the effect and the original source to allow more continuous results at the Mask edges and more detailed control over the warping amounts. Only the alpha channel of this input is used (or green/luma if there is no alpha). This input can optionally be blurred or inverted using the Blur Mask or Invert Mask parameters.

Parameters:

aFrequency: *Default:* 6, *Range:* 0.01 or greater.
The frequency of the first set of waves. Increase for more waves, decrease for fewer.

aAmplitude: *Default:* 0.06, *Range:* any.
The amplitude of the first set of waves.

aAngle: *Default:* 45, *Range:* any.
The rotation angle of the first set of waves in degrees.

aDisplaceAngle: *Default:* 0, *Range:* any.
The warping direction of the first set of waves in degrees relative to their angle.

aPhaseStart: *Default:* 0, *Range:* any.
The phase shift of the first set of waves.

aPhaseSpeed: *Default:* 1, *Range:* any.
If non−zero, the first set of wave automatically travels at this rate.

bFrequency: *Default:* 3, *Range:* 0.01 or greater.
The frequency of the second set of waves. Increase for more waves, decrease for fewer.

bAmplitude: *Default:* 0.12, *Range:* any.
The amplitude of the second set of waves.

bAngle: *Default:* 15, *Range:* any.
The rotation angle of the second set of waves in degrees.

bDisplaceAngle: *Default:* 0, *Range:* any.
The warping direction of the second set of waves in degrees relative to their angle.

bPhaseStart: *Default:* 0, *Range:* any.
The phase shift of the second set of waves.

bPhaseSpeed: *Default:* −1, *Range:* any.

If non−zero, the second set of wave automatically travels at this rate.

zDist: *Default:* 1, *Range:* 0.001 or greater.
Scales the 'distance' of the image. Values greater than 1.0 move it farther away and make it smaller. Values less then 1.0 move the image closer and enlarge it. Zooming in slightly can sometimes be used to hide edge artifacts.

wrap: *X & Y, Radio buttons, Default:* [*Reflect Reflect*].
Determines the method for accessing outside the borders of the source image.

> ***No:*** gives the background pixel color beyond the borders.
> ***Tile:*** repeats a copy of the image.
> ***Reflect:*** repeats a mirrored copy. Edges are often less visible with this method.

filter: *Toggle−button, Default:* on.
If enabled, the image is adaptively filtered when it is resampled. This gives a better quality result when parts of the image are warped smaller.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

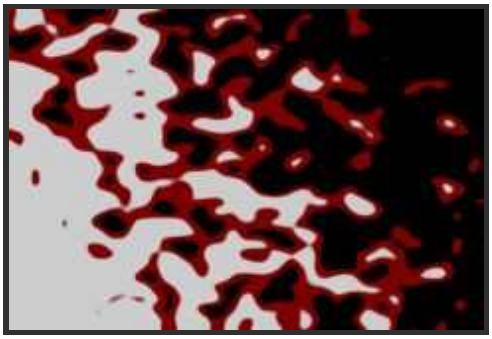
invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

Script Form: `S_Warps(Source, S_Mask, 1.06, "Warp Waves2", aFrequency, etc...);`

See Also:

WarpTransform
WarpVortex
WarpPerspective
WarpWaves
WarpPuddle
WarpBubble
WarpBubble2
WarpFishEye
WarpPuff
WarpPolar
WarpRepeat
WarpChroma
DissolveWaves
Sapphire Plug−ins Introduction

WipeBlobs



In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Performs a wipe transition between two input clips using a pattern of blobs generated by a noise function. The Wipe Amount parameter should be animated to control the transition speed. Increase the Grad Add parameter to make the timing of the blobs pattern move across the screen during the wipe. Increase the Border Width parameter to draw a border at the wipe transition edges.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeSoftness: *Default:* 0, *Range:* 0 or greater.
The width of the transition edges. Larger values will cause softer, less visible edges in the wipe pattern.

frequency: *Default:* 6, *Range:* 0.1 or greater.
The frequency of the blobs pattern. Increase for more and smaller blobs, or decrease for fewer and larger.

relWidth: *Default:* 1, *Range:* 0.1 or greater.
The relative horizontal size of the blobs. Increase for wider blobs, decrease for taller ones.

octaves: *Integer*, *Default:* 1, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

seed: *Default:* 0.432, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

shift_: *X & Y*, *Default:* [.5*width .5*height], *Range:* any.
Translation of the blobs pattern.

borderWidth: *Default:* 0, *Range:* 0 or greater.
If positive, a colored border is drawn at the wipe transition edges, using the border color, opacity, softness, and shift parameters below.

borderOpacity: *Default:* 1, *Range:* 0 to 1.
The opacity of the border. Decrease to make the border transparent and allow the image under it to show through. This has no effect unless Border Width is positive.

borderSoftness: *Default:* 0, *Range:* 0 or greater.
The softness of the border edges. This has no effect unless Border Width is positive.

borderShift: *Default:* 0, *Range:* any.

Shifts the border ahead of or behind the transition edge. This has no effect unless Border Width is positive.

borderColor: *Default rgb:* [0.75 0.75 0.75].
The color of the border. This has no effect unless Border Width is positive.

gradAdd: *Default:* 0, *Range:* 10 or less.
If positive, a gradient will be added to the timing of the transition pattern so it moves across the screen during the wipe. This parameter can be adjusted using the Wipe Widget if enabled, but the value must be positive to make this widget visible.

gradAngle: *Default:* 0, *Range:* any.
The direction of the wipe gradient in counter−clockwise degrees. This will have no effect unless Grad Add is positive. The Wipe Widget also allows adjusting this parameter.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Wipes(From, To, 1.06, "Wipe Blobs", mixPercent, etc...);`

See Also:

WipeLine
WipeCircle
WipeRectangle
WipeStar
WipeClock
WipeWedge
WipeDoubleWedge
WipeFourWedges
WipeDots
WipeChecker
WipeStripes
WipeRings
WipeCells
WipeTiles
WipePixelate
WipeDiffuse
WipeBubble
WipeClouds

Clouds
DissolveSpeckle
Sapphire Plug−ins Introduction

WipeBubble

In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Wipes between two input clips with a bubble−warp process performed within the transition area. The Wipe Amount parameter should be animated to control the transition speed.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeWidth: *Default:* .75*width, *Range:* 5 or greater.
The width of the transition area. This can be adjusted using the Wipe Widget.

angle: *Default:* 0, *Range:* any.
The angle of the wipe direction in counter−clockwise degrees from the right. This can be adjusted using the Wipe Widget.

bubbleAmount: *Default:* 0.5, *Range:* 0 or greater.
The magnitude of the bubble distortion.

frequency: *Default:* 8, *Range:* 0.1 or greater.
The frequency of the bubble pattern. Increase to zoom out, decrease to zoom in.

frequencyRelX: *Default:* 1, *Range:* 0.01 or greater.
The relative horizontal frequency of the bubble pattern. Increase for taller bubbles, decrease for wider bubbles.

octaves: *Integer*, *Default:* 8, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

seed: *Default:* 0.23, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

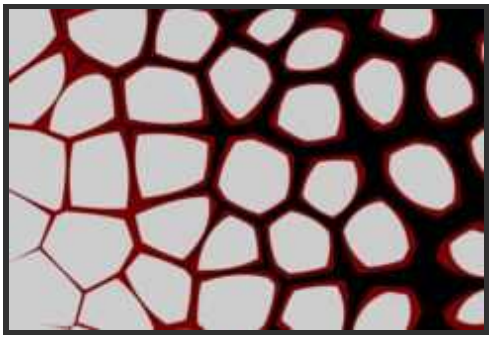
Script Form: `S_Wipes(From, To, 1.06, "Wipe Bubble", mixPercent, etc...);`

See Also:

WipeLine
WipeCircle
WipeRectangle
WipeStar
WipeClock
WipeWedge
WipeDoubleWedge
WipeFourWedges
WipeDots
WipeChecker
WipeStripes
WipeRings
WipeBlobs
WipeCells
WipeTiles
WipePixelate
WipeDiffuse
WipeClouds

WarpBubble
DissolveBubble
Sapphire Plug−ins Introduction

WipeCells



In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Performs a wipe transition between two input clips using a pattern of procedurally generated cellular shapes. The Wipe Amount parameter should be animated to control the transition speed. Increase the Grad Add parameter to make the timing of the cells pattern move across the screen during the wipe. Increase the Border Width parameter to draw a border at the wipe transition edges.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeSoftness: *Default:* 0, *Range:* 0 or greater.
The width of the transition edges. Larger values will cause softer, less visible edges in the wipe pattern.

cells: *Radio buttons, Default: Grow.*
The direction of the cells transition.

Shrink: the cells start large and shrink inwards.
Grow: the cells start small and grow outwards.

frequency: *Default:* 6, *Range:* 0.1 or greater.
The frequency of the cells pattern. Increase for more and smaller cells, or decrease for fewer and larger.

relWidth: *Default:* 1, *Range:* 0.1 or greater.
The relative horizontal size of the cells. Increase for wider cells, decrease for taller ones.

seed: *Default:* 0.432, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

shift_: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translation of the cells pattern.

borderWidth: *Default:* 0, *Range:* 0 or greater.
If positive, a colored border is drawn at the wipe transition edges, using the border color, opacity, softness, and shift parameters below.

borderOpacity: *Default:* 1, *Range:* 0 to 1.
The opacity of the border. Decrease to make the border transparent and allow the image under it to show through. This has no effect unless Border Width is positive.

borderSoftness: *Default:* 0, *Range:* 0 or greater.
The softness of the border edges. This has no effect unless Border Width is positive.

borderShift: *Default:* 0, *Range:* any.
Shifts the border ahead of or behind the transition edge. This has no effect unless Border Width is positive.

borderColor: *Default rgb:* [0.75 0.75 0.75].
The color of the border. This has no effect unless Border Width is positive.

gradAdd: *Default:* 0, *Range:* 10 or less.
If positive, a gradient will be added to the timing of the transition pattern so it moves across the screen during the wipe. This parameter can be adjusted using the Wipe Widget if enabled, but the value must be positive to make this widget visible.

gradAngle: *Default:* 0, *Range:* any.
The direction of the wipe gradient in counter−clockwise degrees. This will have no effect unless Grad Add is positive. The Wipe Widget also allows adjusting this parameter.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

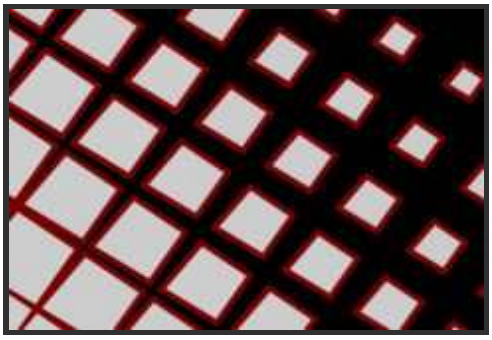
Script Form: `S_Wipes(From, To, 1.06, "Wipe Cells", mixPercent, etc...);`

See Also:

WipeLine
WipeCircle
WipeRectangle
WipeStar
WipeClock
WipeWedge
WipeDoubleWedge
WipeFourWedges
WipeDots
WipeChecker
WipeStripes
WipeRings
WipeBlobs
WipeTiles
WipePixelate
WipeDiffuse
WipeBubble
WipeClouds

TextureCells
Sapphire Plug−ins Introduction

WipeChecker



In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Performs a wipe transition between two input clips using a grid of growing or shrinking checkers. The Wipe Amount parameter should be animated to control the transition speed. Increase the Grad Add parameter to make the timing of the checker pattern move across the screen during the wipe. Increase the Border Width parameter to draw a border at the wipe transition edges.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeSoftness: *Default:* 0, *Range:* 0 or greater.
The width of the transition edges. Larger values will cause softer, less visible edges in the wipe pattern.

checkers: *Radio buttons, Default: Grow*.
The direction of the checkers transition.

Shrink: the squares start large and shrink inwards.
Grow: the squares start small and grow outwards.

angle: *Default:* 45, *Range:* any.
The rotation of the overall checker pattern used for the wipe, in counter−clockwise degrees.

frequency: *Default:* 6, *Range:* 0.1 or greater.
The frequency of the checker pattern. Increase for more and smaller checker, or decrease for fewer and larger.

relWidth: *Default:* 1, *Range:* 0.1 or greater.
The relative horizontal size of the checkers. Increase for wider checkers, decrease for taller ones.

relWidPreRot: *Default:* 1, *Range:* 0.1 or greater.
The relative size of the checkers in the direction of the current rotation angle. If the Angle parameter is zero this will have the same effect as Rel Width.

shift_: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translation of the checker pattern.

borderWidth: *Default:* 0, *Range:* 0 or greater.
If positive, a colored border is drawn at the wipe transition edges, using the border color, opacity, softness, and shift parameters below.

borderOpacity: *Default:* 1, *Range:* 0 to 1.
The opacity of the border. Decrease to make the border transparent and allow the image under it to show through. This has no effect unless Border Width is positive.

borderSoftness: *Default:* 0, *Range:* 0 or greater.
The softness of the border edges. This has no effect unless Border Width is positive.

borderShift: *Default:* 0, *Range:* any.
Shifts the border ahead of or behind the transition edge. This has no effect unless Border Width is positive.

borderColor: *Default rgb:* [0.75 0.75 0.75].
The color of the border. This has no effect unless Border Width is positive.

gradAdd: *Default:* 0, *Range:* 10 or less.
If positive, a gradient will be added to the timing of the transition pattern so it moves across the screen during the wipe. This parameter can be adjusted using the Wipe Widget if enabled, but the value must be positive to make this widget visible.

gradAngle: *Default:* 0, *Range:* any.
The direction of the wipe gradient in counter−clockwise degrees. This will have no effect unless Grad Add is positive. The Wipe Widget also allows adjusting this parameter.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

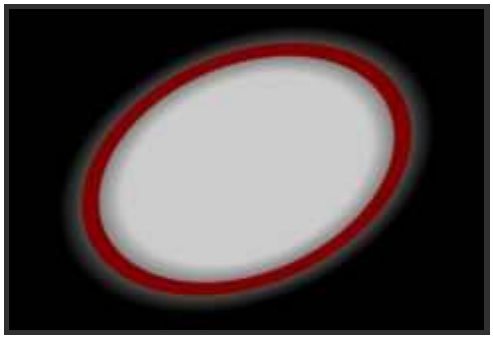
Script Form: `S_Wipes(From, To, 1.06, "Wipe Checker", mixPercent, etc...);`

See Also:

WipeLine
WipeCircle
WipeRectangle
WipeStar
WipeClock
WipeWedge
WipeDoubleWedge
WipeFourWedges
WipeDots
WipeStripes
WipeRings
WipeBlobs
WipeCells
WipeTiles
WipePixelate
WipeDiffuse
WipeBubble
WipeClouds

Grid
Sapphire Plug−ins Introduction

WipeCircle



In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Performs a wipe transition between two input clips using a growing or shrinking circle. The Wipe Amount parameter should be animated to control the transition speed. Increase the Border Width parameter to draw a border at the wipe transition edges.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeSoftness: *Default:* 0, *Range:* 0 or greater.
The width of the transition edges. Larger values will cause softer, less visible edges in the wipe pattern.

wipeDirection: *Radio buttons, Default: In.*
The direction of the circle wipe.

In: the circle contains the first image and shrinks inwards.
Out: the circle contains the second image and grows outwards.

relWidth: *Default:* 1, *Range:* 0.1 or greater.
The relative width of the 'circle' shape. Increase to make a wider oval, decrease to make a taller one.

rotate: *Default:* 0, *Range:* any.
The rotation angle of the 'circle' in counter−clockwise degrees. This has no effect if the Rel Width parameter is 1.0.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The location of the circle center in screen coordinates relative to the center of the frame. This parameter can be set by enabling and moving the Center Widget. Note that moving the circle center can also cause the circle size to change so that the current value of Wipe Amt remains correct.

borderWidth: *Default:* 0, *Range:* 0 or greater.
If positive, a colored border is drawn at the wipe transition edges, using the border color, opacity, softness, and shift parameters below.

borderOpacity: *Default:* 1, *Range:* 0 to 1.
The opacity of the border. Decrease to make the border transparent and allow the image under it to show through. This has no effect unless Border Width is positive.

borderSoftness: *Default:* 0, *Range:* 0 or greater.
The softness of the border edges. This has no effect unless Border Width is positive.

borderShift: *Default:* 0, *Range:* any.
Shifts the border ahead of or behind the transition edge. This has no effect unless Border Width is positive.

borderColor: *Default rgb:* [0.75 0.75 0.75].
The color of the border. This has no effect unless Border Width is positive.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

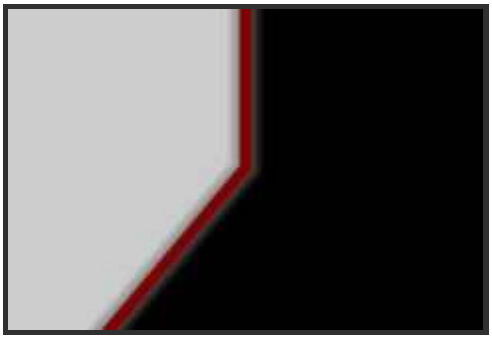
Script Form: `S_Wipes(From, To, 1.06, "Wipe Circle", mixPercent, etc...);`

See Also:

WipeLine
WipeRectangle
WipeStar
WipeClock
WipeWedge
WipeDoubleWedge
WipeFourWedges
WipeDots
WipeChecker
WipeStripes
WipeRings
WipeBlobs
WipeCells
WipeTiles
WipePixelate
WipeDiffuse
WipeBubble
WipeClouds

Sapphire Plug−ins Introduction

WipeClock



In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Performs a clock wipe transition between two input clips. The Wipe Amount parameter should be animated to control the transition speed. Increase the Border Width parameter to draw a border at the wipe transition edges.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeSoftness: *Default:* 0, *Range:* 0 or greater.
The width of the transition edges. Larger values will cause softer, less visible edges in the wipe pattern.

direction: *Radio buttons, Default: Clockwise.*
Selects the direction of the edge rotation.

Clockwise: wipes with an edge rotating clockwise.
CounterCW: wipes with an edge rotating counter clockwise.

angleOpen: *Default:* 0, *Range:* any.
The angle in degrees at which the wipe opens at the start.

angleClose: *Default:* 0, *Range:* any.
The angle in degrees at which the wipe closes at the finish. If Angle Open and Close are not equal, both edges will rotate. For example for a double edged clock wipe set Angle Close to 180.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The location of the clock center in screen coordinates relative to the center of the frame. This parameter can be set by enabling and moving the Center Widget.

borderWidth: *Default:* 0, *Range:* 0 or greater.
If positive, a colored border is drawn at the wipe transition edges, using the border color, opacity, softness, and shift parameters below.

borderOpacity: *Default:* 1, *Range:* 0 to 1.
The opacity of the border. Decrease to make the border transparent and allow the image under it to show through. This has no effect unless Border Width is positive.

borderSoftness: *Default:* 0, *Range:* 0 or greater.
The softness of the border edges. This has no effect unless Border Width is positive.

borderShift: *Default:* 0, *Range:* any.
Shifts the border ahead of or behind the transition edge. This has no effect unless Border Width is positive. For the clock wipe pattern, the shift amount is limited to within the area of Edge Softness.

borderColor: *Default rgb:* [0.75 0.75 0.75].
The color of the border. This has no effect unless Border Width is positive.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Wipes(From, To, 1.06, "Wipe Clock", mixPercent, etc...);`

See Also:

WipeLine
WipeCircle
WipeRectangle
WipeStar
WipeWedge
WipeDoubleWedge
WipeFourWedges
WipeDots
WipeChecker
WipeStripes
WipeRings
WipeBlobs
WipeCells
WipeTiles
WipePixelate
WipeDiffuse
WipeBubble
WipeClouds

Sapphire Plug−ins Introduction

WipeClouds

In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Transitions from the first clip to the second using a moving cloud texture. The Wipe Amount parameter should be animated to control the transition speed.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

frequency: *Default:* 2, *Range:* 0.1 or greater.
The frequency of the clouds pattern. Increase for more and smaller clouds, or decrease for fewer and larger.

frequencyRelX: *Default:* 0.4, *Range:* 0.01 or greater.
The relative horizontal frequency of the texture. Increase to stretch it vertically or decrease to stretch it horizontally.

octaves: *Integer, Default:* 8, *Range:* 1 to 10.
The number of summed layers of noise. Each octave is twice the frequency and half the amplitude of the previous. A single octave gives a smooth texture. Adding octaves makes the result approach a fractal (1/f) noise texture.

seed: *Default:* 0.23, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

shiftStart: *X & Y, Default:* [0 0], *Range:* any.
Translation offset of the texture. Since the texture is procedurally generated it can be shifted with no repeating units or seams occurring.

shiftSpeed: *X & Y, Default:* [width 0], *Range:* any.
Translation speed of the texture. If non−zero, the result is automatically animated to shift at this rate. The result of animated Speed values may not be intuitive, so for variable speed motion it is usually best to set this to 0 and animate the Shift Start values instead.

gradAdd: *Default:* 0, *Range:* 10 or less.
If positive, a gradient will be added to the timing of the transition pattern so it moves across the screen during the wipe. This parameter can be adjusted using the Wipe Widget if enabled, but the value must be positive to make this widget visible.

gradAngle: *Default:* 0, *Range:* any.
The direction of the wipe gradient in counter−clockwise degrees. This will have no effect unless Grad Add is positive. The Wipe Widget also allows adjusting this parameter.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Wipes(From, To, 1.06, "Wipe Clouds", mixPercent, etc...);`

See Also:

WipeLine
WipeCircle
WipeRectangle
WipeStar
WipeClock
WipeWedge
WipeDoubleWedge
WipeFourWedges
WipeDots
WipeChecker
WipeStripes
WipeRings
WipeBlobs
WipeCells
WipeTiles
WipePixelate
WipeDiffuse
WipeBubble

Clouds
Sapphire Plug−ins Introduction

WipeDiffuse

In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Wipes between two input clips with a pixel−diffusion process performed within the transition area. The Wipe Amount parameter should be animated to control the transition speed. The pixelated look of this effect depends on the image resolution, so it is recommended to test your final resolution before processing.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeWidth: *Default:* .75*width, *Range:* 5 or greater.
The width of the transition area. This can be adjusted using the Wipe Widget.

angle: *Default:* 0, *Range:* any.
The angle of the wipe direction in counter−clockwise degrees from the right. This can be adjusted using the Wipe Widget.

diffuseAmount: *Default:* .1*width, *Range:* 0 or greater.
The magnitude of the pixel diffusion.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

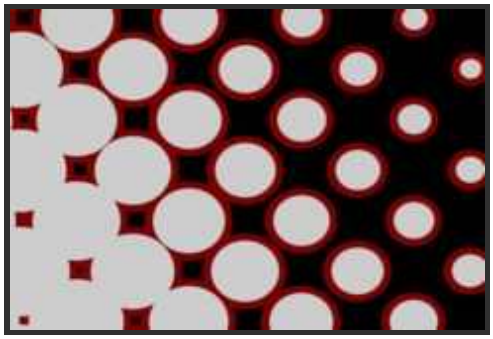
clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Wipes(From, To, 1.06, "Wipe Diffuse", mixPercent, etc...);`

See Also:

WipeLine
WipeCircle
WipeRectangle
WipeStar
WipeClock
WipeWedge
WipeDoubleWedge
WipeFourWedges
WipeDots
Diffuse
DissolveDiffuse
Sapphire Plug−ins Introduction

WipeDots



In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Performs a wipe transition between two input clips using a grid of growing or shrinking dots. The Wipe Amount parameter should be animated to control the transition speed. Increase the Grad Add parameter to make the timing of the dots pattern move across the screen during the wipe. Increase the Border Width parameter to draw a border at the wipe transition edges.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeSoftness: *Default:* 0, *Range:* 0 or greater.
The width of the transition edges. Larger values will cause softer, less visible edges in the wipe pattern.

dots: *Radio buttons, Default: Grow.*
The direction of the dots transition.

Shrink: the dots start large and shrink inwards.
Grow: the dots start small and grow outwards.

angle: *Default:* 45, *Range:* any.
The rotation of the overall dots pattern used for the wipe, in counter−clockwise degrees.

frequency: *Default:* 6, *Range:* 0.1 or greater.
The frequency of the dots pattern. Increase for more and smaller dots, or decrease for fewer and larger.

relWidth: *Default:* 1, *Range:* 0.1 or greater.
The relative horizontal size of the dots. Increase for wider dots, decrease for taller ones.

relWidPreRot: *Default:* 1, *Range:* 0.1 or greater.
The relative size of the dots in the direction of the current rotation angle. If the Angle parameter is zero this will have the same effect as Rel Width.

shift_: *X & Y, Default:* [.5*width .5*height], *Range:* any.
Translation of the dots pattern.

borderWidth: *Default:* 0, *Range:* 0 or greater.
If positive, a colored border is drawn at the wipe transition edges, using the border color, opacity, softness, and shift parameters below.

borderOpacity: *Default:* 1, *Range:* 0 to 1.
The opacity of the border. Decrease to make the border transparent and allow the image under it to show through. This has no effect unless Border Width is positive.

borderSoftness: *Default:* 0, *Range:* 0 or greater.
The softness of the border edges. This has no effect unless Border Width is positive.

borderShift: *Default:* 0, *Range:* any.
Shifts the border ahead of or behind the transition edge. This has no effect unless Border Width is positive.

borderColor: *Default rgb:* [0.75 0.75 0.75].
The color of the border. This has no effect unless Border Width is positive.

gradAdd: *Default:* 0, *Range:* 10 or less.
If positive, a gradient will be added to the timing of the transition pattern so it moves across the screen during the wipe. This parameter can be adjusted using the Wipe Widget if enabled, but the value must be positive to make this widget visible.

gradAngle: *Default:* 0, *Range:* any.
The direction of the wipe gradient in counter−clockwise degrees. This will have no effect unless Grad Add is positive. The Wipe Widget also allows adjusting this parameter.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

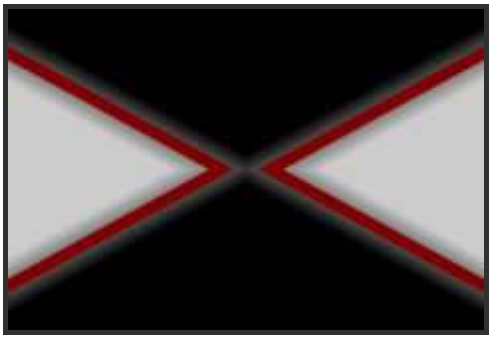
Script Form: `S_Wipes(From, To, 1.06, "Wipe Dots", mixPercent, etc...);`

See Also:

WipeLine
WipeCircle
WipeRectangle
WipeStar
WipeClock
WipeWedge
WipeDoubleWedge
WipeFourWedges
WipeChecker
WipeStripes
WipeRings
WipeBlobs
WipeCells
WipeTiles
WipePixelate
WipeDiffuse
WipeBubble
WipeClouds

HalfTone
Sapphire Plug−ins Introduction

WipeDoubleWedge



In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Performs a wipe transition between two input clips using two wedge shapes. The Wipe Amount parameter should be animated to control the transition speed. Increase the Border Width parameter to draw a border at the wipe transition edges.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeSoftness: *Default:* 0, *Range:* 0 or greater.
The width of the transition edges. Larger values will cause softer, less visible edges in the wipe pattern.

wipeDirection: *Radio buttons, Default: In.*
Selects the direction of the motion of the wedges.

In: the wedge contains the first image and shrinks inwards.
Out: the wedge contains the second image and grows outwards.

angle: *Default:* 0, *Range:* any.
The rotation angle of the wedge shapes in counter−clockwise degrees.

sharpness: *Default:* 2, *Range:* 0 or greater.
The sharpness of the point of the wedges.

borderWidth: *Default:* 0, *Range:* 0 or greater.
If positive, a colored border is drawn at the wipe transition edges, using the border color, opacity, softness, and shift parameters below.

borderOpacity: *Default:* 1, *Range:* 0 to 1.
The opacity of the border. Decrease to make the border transparent and allow the image under it to show through. This has no effect unless Border Width is positive.

borderSoftness: *Default:* 0, *Range:* 0 or greater.
The softness of the border edges. This has no effect unless Border Width is positive.

borderShift: *Default:* 0, *Range:* any.
Shifts the border ahead of or behind the transition edge. This has no effect unless Border Width is positive.

borderColor: *Default rgb:* [0.75 0.75 0.75].
The color of the border. This has no effect unless Border Width is positive.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111

for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

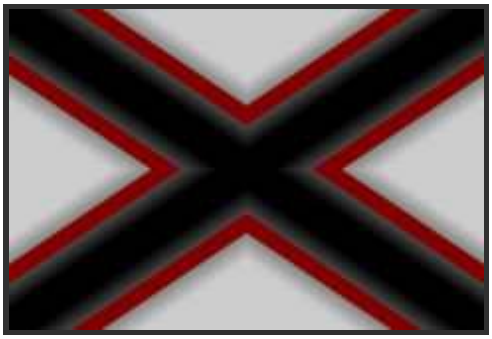
Script Form: `S_Wipes(From, To, 1.06, "Wipe Double Wedge", mixPercent, etc...);`

See Also:

WipeLine
WipeCircle
WipeRectangle
WipeStar
WipeClock
WipeWedge
WipeFourWedges
WipeDots
WipeChecker
WipeStripes
WipeRings
WipeBlobs
WipeCells
WipeTiles
WipePixelate
WipeDiffuse
WipeBubble
WipeClouds

Sapphire Plug−ins Introduction

WipeFourWedges



In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Performs a wipe transition between two input clips using a pattern of four wedges merging into an 'X' shape. The Wipe Amount parameter should be animated to control the transition speed. Increase the Border Width parameter to draw a border at the wipe transition edges.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeSoftness: *Default:* 0, *Range:* 0 or greater.
The width of the transition edges. Larger values will cause softer, less visible edges in the wipe pattern.

wipeDirection: *Radio buttons, Default: In.*
Selects the direction of motion of the wedges.

In: wedges move inwards making an 'X' shape.
Out: starts with an 'X' and widens outwards.

angle: *Default:* 0, *Range:* any.
The rotation angle in counter−clockwise degrees of the wedge pattern.

aspectScale: *Default:* 1, *Range:* 0.01 or greater.
Scales the aspect ratio of the wedge pattern. Increase to stretch the shapes in the horizontal direction.

borderWidth: *Default:* 0, *Range:* 0 or greater.
If positive, a colored border is drawn at the wipe transition edges, using the border color, opacity, softness, and shift parameters below.

borderOpacity: *Default:* 1, *Range:* 0 to 1.
The opacity of the border. Decrease to make the border transparent and allow the image under it to show through. This has no effect unless Border Width is positive.

borderSoftness: *Default:* 0, *Range:* 0 or greater.
The softness of the border edges. This has no effect unless Border Width is positive.

borderShift: *Default:* 0, *Range:* any.
Shifts the border ahead of or behind the transition edge. This has no effect unless Border Width is positive.

borderColor: *Default rgb:* [0.75 0.75 0.75].
The color of the border. This has no effect unless Border Width is positive.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111

for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Wipes(From, To, 1.06, "Wipe Four Wedges", mixPercent, etc...);`

See Also:

WipeLine
WipeCircle
WipeRectangle
WipeStar
WipeClock
WipeWedge
WipeDoubleWedge
WipeDots
WipeChecker
WipeStripes
WipeRings
WipeBlobs
WipeCells
WipeTiles
WipePixelate
WipeDiffuse
WipeBubble
WipeClouds

Sapphire Plug−ins Introduction

WipeLine

In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Performs a simple line wipe transition between two input clips. The Wipe Amount parameter should be animated to control the transition speed. Increase the Border Width parameter to draw a border at the wipe transition edges.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeSoftness: *Default:* 0, *Range:* 0 or greater.
The width of the transition edges. Larger values will cause softer, less visible edges in the wipe pattern.

angle: *Default:* 45, *Range:* any.
The angle of the wipe direction in degrees. Use 0 for a wipe from left to right, 90 or −90 for a vertical wipe, 180 for a wipe from right to left.

borderWidth: *Default:* 0, *Range:* 0 or greater.
If positive, a colored border is drawn at the wipe transition edges, using the border color, opacity, softness, and shift parameters below.

borderOpacity: *Default:* 1, *Range:* 0 to 1.
The opacity of the border. Decrease to make the border transparent and allow the image under it to show through. This has no effect unless Border Width is positive.

borderSoftness: *Default:* 0, *Range:* 0 or greater.
The softness of the border edges. This has no effect unless Border Width is positive.

borderShift: *Default:* 0, *Range:* any.
Shifts the border ahead of or behind the transition edge. This has no effect unless Border Width is positive.

borderColor: *Default rgb:* [0.75 0.75 0.75].
The color of the border. This has no effect unless Border Width is positive.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

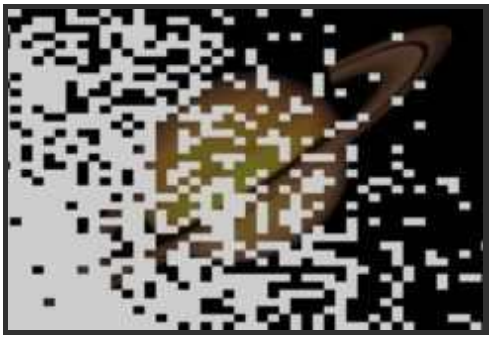
Script Form: `S_Wipes(From, To, 1.06, "Wipe Line", mixPercent, etc...);`

See Also:

WipeCircle
WipeRectangle
WipeStar
WipeClock
WipeWedge
WipeDoubleWedge
WipeFourWedges
WipeDots
WipeChecker
WipeStripes
WipeRings
WipeBlobs
WipeCells
WipeTiles
WipePixelate
WipeDiffuse
WipeBubble
WipeClouds

Gradient
Sapphire Plug−ins Introduction

WipePixelate



In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Transitions between two input clips by adding blocks of pixels of one clip onto another in a semi−random order. The Wipe Amount parameter should be animated to control the transition speed. Adjust the Edge Width and Chunky parameters for different pixelated patterns.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeWidth: *Default:* .5*width, *Range:* 5 or greater.
The width of the transition area.

angle: *Default:* 0, *Range:* any.
The angle of the wipe direction in degrees. Use 0 for a wipe from left to right, 90 or −90 for a vertical wipe, 180 for a wipe from right to left.

pixelFrequency: *Default:* 20, *Range:* 0.1 or greater.
Increase for smaller and more pixels, decrease for fewer and larger pixels.

pixelRelWidth: *Default:* 1, *Range:* 0.01 or greater.
The relative horizontal size of the pixels. Increase for wide pixels, decrease for tall ones.

chunky: *Default:* 0, *Range:* 0 or greater.
Increase to cause the pixels to be added with a more clustered ordering.

seed: *Default:* 0.23, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different results and the same value should give a repeatable result.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

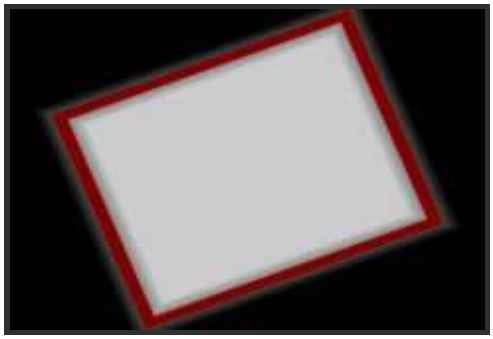
Script Form: `S_Wipes(From, To, 1.06, "Wipe Pixelate", mixPercent, etc...);`

See Also:

WipeLine
WipeCircle
WipeRectangle
WipeStar
WipeClock
WipeWedge
WipeDoubleWedge
WipeFourWedges
WipeDots
WipeChecker
WipeStripes
WipeRings
WipeBlobs
WipeCells
WipeTiles
WipeDiffuse
WipeBubble
WipeClouds

Mosaic
Sapphire Plug−ins Introduction

WipeRectangle



In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Performs a wipe transition between two input clips using a growing or shrinking rectangle. The Wipe Amount parameter should be animated to control the transition speed. Increase the Border Width parameter to draw a border at the wipe transition edges.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeSoftness: *Default:* 0, *Range:* 0 or greater.
The width of the transition edges. Larger values will cause softer, less visible edges in the wipe pattern.

wipeDirection: *Radio buttons, Default: In.*
The direction of the rectangle wipe.

In: the rectangle contains the first image and shrinks inwards.
Out: the rectangle contains the second image and grows outwards.

angle: *Default:* 0, *Range:* any.
The rotation angle of the rectangle in counter−clockwise degrees.

relWidth: *Default:* 1.25, *Range:* 0.1 or greater.
The relative width of the rectangle. Increase to make wider, decrease to make thinner.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The location of the rectangle center in screen coordinates relative to the center of the frame. This parameter can be set by enabling and moving the Center Widget. Note that moving the rectangle center can also cause the rectangle size to change so that the current value of Wipe Amt remains correct.

borderWidth: *Default:* 0, *Range:* 0 or greater.
If positive, a colored border is drawn at the wipe transition edges, using the border color, opacity, softness, and shift parameters below.

borderOpacity: *Default:* 1, *Range:* 0 to 1.
The opacity of the border. Decrease to make the border transparent and allow the image under it to show through. This has no effect unless Border Width is positive.

borderSoftness: *Default:* 0, *Range:* 0 or greater.
The softness of the border edges. This has no effect unless Border Width is positive.

borderShift: *Default:* 0, *Range:* any.
Shifts the border ahead of or behind the transition edge. This has no effect unless Border Width is positive.

borderColor: *Default rgb:* [0.75 0.75 0.75].
The color of the border. This has no effect unless Border Width is positive.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Wipes(From, To, 1.06, "Wipe Rectangle", mixPercent, etc...);`

See Also:

WipeLine
WipeCircle
WipeStar
WipeClock
WipeWedge
WipeDoubleWedge
WipeFourWedges
WipeDots
WipeChecker
WipeStripes
WipeRings
WipeBlobs
WipeCells
WipeTiles
WipePixelate
WipeDiffuse
WipeBubble
WipeClouds

Sapphire Plug−ins Introduction

WipeStar

In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Performs a wipe transition between two input clips using a star shape. The Wipe Amount parameter should be animated to control the transition speed. Increase the Border Width parameter to draw a border at the wipe transition edges.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeSoftness: *Default:* 0, *Range:* 0 or greater.
The width of the transition edges. Larger values will cause softer, less visible edges in the wipe pattern.

wipeDirection: *Radio buttons, Default: In.*
The direction of the star wipe.

In: the star contains the first image and shrinks inwards.
Out: the star contains the second image and grows outwards.

points: *Integer, Default:* 5, *Range:* 3 or greater.
The number of points in the star.

pointiness: *Default:* 1.1, *Range:* 0 or greater.
The pointiness of the star. Increase for sharp spikes, decrease for more regular polygonal shapes.

angle: *Default:* 0, *Range:* any.
The rotation angle of the star in counter−clockwise degrees.

relWidth: *Default:* 1, *Range:* 0.1 or greater.
The relative horizontal size of the star. Increase for wider star, decrease for taller ones.

center: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The location of the star center in screen coordinates relative to the center of the frame. This parameter can be set by enabling and moving the Center Widget. Note that moving the star center can also cause the star size to change so that the current value of Wipe Amt remains correct.

borderWidth: *Default:* 0, *Range:* 0 or greater.
If positive, a colored border is drawn at the wipe transition edges, using the border color, opacity, softness, and shift parameters below.

borderOpacity: *Default:* 1, *Range:* 0 to 1.
The opacity of the border. Decrease to make the border transparent and allow the image under it to show through. This has no effect unless Border Width is positive.

borderSoftness: *Default:* 0, *Range:* 0 or greater.
The softness of the border edges. This has no effect unless Border Width is positive.

borderShift: *Default:* 0, *Range:* any.
Shifts the border ahead of or behind the transition edge. This has no effect unless Border Width is positive.

borderColor: *Default rgb:* [0.75 0.75 0.75].
The color of the border. This has no effect unless Border Width is positive.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Wipes(From, To, 1.06, "Wipe Star", mixPercent, etc...);`

See Also:

WipeLine
WipeCircle
WipeRectangle
WipeClock
WipeWedge
WipeDoubleWedge
WipeFourWedges
WipeDots
WipeChecker
WipeStripes
WipeRings
WipeBlobs
WipeCells
WipeTiles
WipePixelate
WipeDiffuse
WipeBubble
WipeClouds

Sapphire Plug−ins Introduction

WipeStripes

In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Performs a wipe transition between two input clips using a series of stripes. The Wipe Amount parameter should be animated to control the transition speed. Increase the Grad Add parameter to make the timing of the stripe pattern move across the screen during the wipe. Increase the Border Width parameter to draw a border at the wipe transition edges.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeSoftness: *Default:* 0, *Range:* 0 or greater.
The width of the transition edges. Larger values will cause softer, less visible edges in the wipe pattern.

angle: *Default:* 45, *Range:* any.
The rotation of the overall stripes pattern used for the wipe, in counter−clockwise degrees.

frequency: *Default:* 6, *Range:* 0.1 or greater.
The frequency of the stripes pattern. Increase for more and smaller stripes, or decrease for fewer and larger.

shiftStripes: *Default:* 0, *Range:* any.
Translation of the stripe pattern.

borderWidth: *Default:* 0, *Range:* 0 or greater.
If positive, a colored border is drawn at the wipe transition edges, using the border color, opacity, softness, and shift parameters below.

borderOpacity: *Default:* 1, *Range:* 0 to 1.
The opacity of the border. Decrease to make the border transparent and allow the image under it to show through. This has no effect unless Border Width is positive.

borderSoftness: *Default:* 0, *Range:* 0 or greater.
The softness of the border edges. This has no effect unless Border Width is positive.

borderShift: *Default:* 0, *Range:* any.
Shifts the border ahead of or behind the transition edge. This has no effect unless Border Width is positive.

borderColor: *Default rgb:* [0.75 0.75 0.75].
The color of the border. This has no effect unless Border Width is positive.

gradAdd: *Default:* 0, *Range:* 10 or less.
If positive, a gradient will be added to the timing of the transition pattern so it moves across the screen during the wipe. This parameter can be adjusted using the Wipe Widget if enabled, but the value must be positive to make this widget visible.

gradAngle: *Default:* 0, *Range:* any.
The direction of the wipe gradient in counter−clockwise degrees. This will have no effect unless Grad Add is positive. The Wipe Widget also allows adjusting this parameter.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

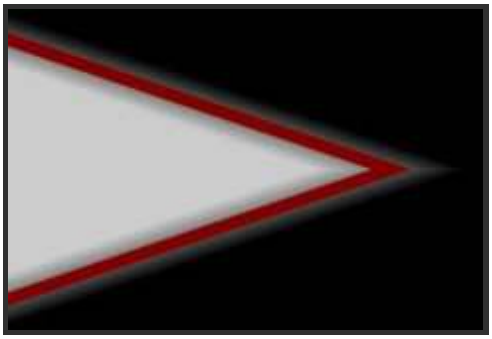
Script Form: `S_Wipes(From, To, 1.06, "Wipe Stripes", mixPercent, etc...);`

See Also:

WipeLine
WipeCircle
WipeRectangle
WipeStar
WipeClock
WipeWedge
WipeDoubleWedge
WipeFourWedges
WipeDots
WipeChecker
WipeRings
WipeBlobs
WipeCells
WipeTiles
WipePixelate
WipeDiffuse
WipeBubble
WipeClouds

Sapphire Plug−ins Introduction

WipeWedge



In the S_Wipes Plugin, and also in the Mixer menu of Shake's Transition node

Performs a wipe transition between two input clips using a wedge shape. The Wipe Amount parameter should be animated to control the transition speed. Increase the Border Width parameter to draw a border at the wipe transition edges.

Inputs:

From: Starts the transition with this clip.

To: Ends the transition with this clip.

Parameters:

mixPercent: *Default:* Transition Curve, *Range:* −10 to 110.
The transition percent between the From and To inputs. A value of 0 gives only the From input and a value of 100 gives only the To input. By default this parameter will automatically animate from 0 to 100 to perform a complete transition.

edgeSoftness: *Default:* 0, *Range:* 0 or greater.
The width of the transition edges. Larger values will cause softer, less visible edges in the wipe pattern.

wipeDirection: *Radio buttons, Default: In.*
Selects the direction of the wedge motion.

In: the wedge contains the first image and shrinks inwards.
Out: the wedge contains the second image and grows outwards.

angle: *Default:* 0, *Range:* any.
The rotation angle of the wedge shape in counter−clockwise degrees.

sharpness: *Default:* 2, *Range:* 0 or greater.
The sharpness of the point of the wedge.

borderWidth: *Default:* 0, *Range:* 0 or greater.
If positive, a colored border is drawn at the wipe transition edges, using the border color, opacity, softness, and shift parameters below.

borderOpacity: *Default:* 1, *Range:* 0 to 1.
The opacity of the border. Decrease to make the border transparent and allow the image under it to show through. This has no effect unless Border Width is positive.

borderSoftness: *Default:* 0, *Range:* 0 or greater.
The softness of the border edges. This has no effect unless Border Width is positive.

borderShift: *Default:* 0, *Range:* any.
Shifts the border ahead of or behind the transition edge. This has no effect unless Border Width is positive.

borderColor: *Default rgb:* [0.75 0.75 0.75].
The color of the border. This has no effect unless Border Width is positive.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111

for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_Wipes(From, To, 1.06, "Wipe Wedge", mixPercent, etc...);`

See Also:

WipeLine
WipeCircle
WipeRectangle
WipeStar
WipeClock
WipeDoubleWedge
WipeFourWedges
WipeDots
WipeChecker
WipeStripes
WipeRings
WipeBlobs
WipeCells
WipeTiles
WipePixelate
WipeDiffuse
WipeBubble
WipeClouds

Sapphire Plug−ins Introduction

ZDepthCueBlur

Blurs areas of the source clip by different amounts using depth values from a ZBuffer input. Separates the input into a number of layers between Z Near and Z Far and blurs them by different amounts depending on Blur Near and Blur Far. Linear fog can also be mixed into the result.

Inputs:

Source: The clip to be processed.

ZBuffer: The input clip containing depth values for each Source pixel. Only the luminance of this input is used. It does not access the Z channel even if one is present. If necessary you can use Shake's Rewire or ColorX to first copy the Z to the luminance. Normally black corresponds to the nearest objects and white to the farthest.

Parameters:

layerMode: *Radio buttons, Default: Interp.*
Determines how the differently blurred layers are combined.

> ***Comp:*** the closer layers are composited over the farther layers. This method often gives better results if you have objects at different depths overlapping each other with discontinuous values in your depth image. However, the seams between layers are sometimes visible with this option.
> ***Interp:*** the blurred layers are interpolated using depth image values. This method gives smoother transitions between layers, and is usually better if there are no sharp changes in your depth image.

zNear: *Default:* 0.33, *Range:* 0 to 1.
The value of the input Z corresponding to near objects. Z values less than this will be blurred by Blur Near.

zFar: *Default:* 0.66, *Range:* 0 to 1.
The value of the input Z corresponding to far objects. Z values greater than this will be blurred by Blur Far.

layers: *Integer, Default:* 3, *Range:* 2 to 50.
The number of depth layers to separate the source into. More layers require more processing but are sometimes needed to avoid visible seams between the layers blurred by different amounts.

blurNear: *Default:* 0, *Range:* any.
The amount to blur the layer at Z Near.

blurFar: *Default:* .05*width, *Range:* any.
The amount to blur the layer at Z Far. Setting either Blur Near or Blur Far to a negative value causes the interpolated blur amounts to be zero between Z Near and Z Far, as if the focal plane were inbetween those depths.

blurRel: *X & Y, Default:* [1 1/GetDefaultAspect()], *Range:* 0 or greater.
The relative horizontal and vertical blur widths. Set Blur Rel X to 0 for a vertical−only blur, or set Blur Rel Y to 0 for a horizontal−only blur.

fogColor: *Default rgb:* [0.5 0.5 0.5].
The fog color should normally match the sky or background color of the source clip. Use gray for mist, brown for smog, blue for underwater, etc.

fogNear: *Default:* 0, *Range:* 0 to 1.
The amount of fog to add to objects at or closer than Z Near.

fogFar: *Default:* 0, *Range:* 0 to 1.
The amount of fog to add to objects at or farther than Z Far.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_ZDepthCueBlur(Source, ZBuffer, 1.06, layerMode, etc...);`

See Also:

ZFogLinear
ZFogExponential
Sapphire Plug−ins Introduction

ZFogExponential

In the S_ZFog Plugin.

Mixes a fog color into the source clip using depth values from a ZBuffer input. The fog starts at Z Near and increases exponentially to Z Far at a rate depending on the Fog Density.

Inputs:

Source: The clip to be processed.

ZBuffer: The input clip containing depth values for each Source pixel. Only the luminance of this input is used. It does not access the Z channel even if one is present. If necessary you can use Shake's Rewire or ColorX to first copy the Z to the luminance. Normally black corresponds to the nearest objects and white to the farthest.

Parameters:

zNear: *Default:* 0.1, *Range:* 0 to 1.
The value of the input Z corresponding to near objects.

zFar: *Default:* 1, *Range:* 0 to 1.
The value of the input Z corresponding to far objects.

fogColor: *Default rgb:* [0.5 0.5 0.5].
The fog color should normally match the sky or background color of the source clip. Use gray for mist, brown for smog, blue for underwater, etc.

fogDensity: *Default:* 0.7, *Range:* 0 to 1.
The density of the fog.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_ZFog(Source, ZBuffer, 1.06, 1, zNear, etc...);`

See Also:

ZFogLinear ZDepthCueBlur
Sapphire Plug−ins Introduction

ZFogLinear

In the S_ZFog Plugin.

Mixes a fog color into the source clip using depth values from a ZBuffer input. The fog amount varies linearly between Fog Near and Fog Far as the depth varies between Z Near and Z Far.

Inputs:

Source: The clip to be processed.

ZBuffer: The input clip containing depth values for each Source pixel. Only the luminance of this input is used. It does not access the Z channel even if one is present. If necessary you can use Shake's Rewire or ColorX to first copy the Z to the luminance. Normally black corresponds to the nearest objects and white to the farthest.

Parameters:

zNear: *Default:* 0.1, *Range:* 0 to 1.
The value of the input Z corresponding to near objects.

zFar: *Default:* 1, *Range:* 0 to 1.
The value of the input Z corresponding to far objects.

fogColor: *Default rgb:* [0.5 0.5 0.5].
The fog color should normally match the sky or background color of the source clip. Use gray for mist, brown for smog, blue for underwater, etc.

fogNear: *Default:* 0, *Range:* 0 to 1.
The amount of fog to add to objects at or closer than Z Near.

fogFar: *Default:* 0.8, *Range:* 0 to 1.
The amount of fog to add to objects at or farther than Z Far.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_ZFog(Source, ZBuffer, 1.06, 0, zNear, etc...);`

See Also:

ZFogExponential ZDepthCueBlur
Sapphire Plug−ins Introduction

Zap

Generates lightning bolts between two points, and renders them over a background. Increase the number of bolts to give a electrical plasma effect. Increase Vary Endpoint to spread out the ends of the bolts. Adjust the Glow Color for differently colored results. The Wiggle Speed parameter causes the bolts to automatically undulate over time.

Inputs:

Background: The clip to use as background.

Parameters:

bolts: *Integer, Default:* 1, *Range:* 1 to 500.
The number of lightning bolts to draw, each between the Start and End location.

start: *X & Y, Default:* [.25*width .9*height], *Range:* any.
The starting point of the bolts.

end: *X & Y, Default:* [.75*width .1*height], *Range:* any.
The end point of the bolts. This parameter can be adjusted using the End Widget.

varyEndpoint: *Default:* 0, *Range:* 0 or greater.
Offsets the End location by a random amount within a circle of this radius. If Bolts is greater than 1, this can be useful to spread out the different End points. For example, you can create multiple radiating bolts by increasing this radius and placing the End point near the Start point. This parameter can also be adjusted using the End Widget, after it is made positive.

boltWidth: *Default:* .00625*width, *Range:* 0 or greater.
The width of the lightning bolts.

varyWidth: *Default:* 0, *Range:* 0 to 1.
The amount of random variation in the width of the bolts along their lengths.

endPointiness: *Default:* 0.1, *Range:* 0 to 1.
Determines how pointed the end of the bolts are. If 0, the entire bolt will have equal width. If 1, the bolts will thin out along their entire length for a pointed end. If it is .5, the bolts will start thinning out half way between the start and end points.

wiggleStart: *Default:* 0, *Range:* 0 or greater.
By default the bolts automatically wiggle over time. This parameter provides a starting offset for these bolt perturbations.

wiggleSpeed: *Default:* 1, *Range:* 0 or greater.
The speed at which the bolts are perturbed automatically over time. To animate changes in speed, set this to zero and animate the Wiggle Start parameter instead.

jitterFrames: *Integer, Default:* 0, *Range:* 0 or greater.
If this is 0, the same random lightning will be used for every frame processed. If it is 1, different random lightning is used for each frame. If it is 2, new random lightning is used for every other frame, and so on.

randSeed: *Default:* 0.123, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different random lightning bolts, and the same value should give a repeatable result.

wrinkleAmp: *Default:* 1, *Range:* 0 or greater.

Scales the amount of wrinkles in the bolts. Decrease for straighter smoother bolts or increase for more kinky bolts.

curveAmp: *Default:* 0.5, *Range:* 0 or greater.
Similar to Wrinkle Amp but affects the general path of the bolt. If decreased, the bolt will stay closer to the line between the Start and End points. If increased it can wander further away from this line. This differs from the Wrinkle Amp parameter in that it can be used to make straighter bolts while still keeping the wrinkles at the detailed level.

branchiness: *Default:* 1, *Range:* 0 to 20.
Scales the number of additional bolts that branch from the main bolt. Set this to 0 for basic bolts with no extra branches.

branchAngle: *Default:* 65, *Range:* 0 to 180.
The maximum angle of the random branches relative to the bolt they are branching off of. If this is 0 the branches will be more lined up with the main bolt. With larger values the branches will be more perpendicular to the main bolt.

branchLength: *Default:* 0.5, *Range:* 0 to 1.
The maximum length of the branches relative to the distance between the Start and End points.

glowColor: *Default rgb:* [0.5 0.5 1].
The color of the glow applied to the lightning.

glowBright: *Default:* 2, *Range:* 0 or greater.
Scales the brightness of the glow applied to the lightning.

glowWidth: *Default:* .025*width, *Range:* 0 or greater.
The width of the glow applied to the lightning.

glowWidthRed: *Default:* 0.5, *Range:* 0 or greater.
The relative red width of the glow.

glowWidthGrn: *Default:* 1, *Range:* 0 or greater.
The relative green width of the glow.

glowWidthBlue: *Default:* 1.5, *Range:* 0 or greater.
The relative blue width of the glow.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the lightning and its glow. The maximum of the red, green, and blue brightness is scaled by this value and combined with the background Alpha at each pixel.

zapColor: *Default rgb:* [1 1 1].
The color of the lightning. If you want to keep the lightning bolt itself bright white, you can still affect the perceived color by adjusting the Glow Color instead.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the lightning bolts.

startOffset: *Default:* 0, *Range:* 0 to 1.
The offset from the start point to begin drawing the bolts. This can be useful for animating a lightning strike.

length: *Default:* 1, *Range:* 0 to 1.
The length of the bolts, beginning at Start Offset. If less than 1, the bolts will not be drawn all the way from start to end. This can be useful for animating a lightning strike.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background before combining with the lightning. If 0, the result will contain only the

lightning image over black.

combine: *Radio buttons, Default: Screen.*
Determines how the lightning and glow are combined with the Background.

> ***Screen:*** performs a blend function which can help prevent overly bright results.
> ***Add:*** causes the lightning to be added to the background. This gives brighter glows over light backgrounds.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

Script Form: `S_Zap(Background, 1.06, bolts, etc...);`

See Also:

ZapTo
ZapFrom
Sapphire Plug−ins Introduction

ZapFrom

Generates multiple lightning bolts outwards from the edges of objects in the FromObj input clip, and renders them over a background input. Use the Show:Edges option to view the source edges while adjusting the Threshold and Blur From Obj parameters.

Inputs:

FromObject: The edges of objects in this clip are extracted, and the lightning starts at points along these edges.

Background: *Optional.* The clip to use as background.

S_Mask: *Optional.* The lengths of the bolts in each area are scaled by this input. White areas generate normal bolts, gray areas generate shorter bolts, and black areas cause no bolts to be made. Only the alpha channel of this input is used (or green/luma if there is no alpha).

Parameters:

surfaceBolts: *Integer, Default:* 25, *Range:* 1 to 2000.
The number of points along the edges to generate lightning bolts from. These surface bolts are divided up amongst the isolated shapes in the FromObj input, proportionally to the sizes of the shapes.

threshold: *Default:* 0.5, *Range:* 0 to 1.
The value used to determine the edge locations. Objects darker than this value are ignored. On smooth objects, larger threshold values move the edges inwards to make smaller shapes, and smaller values move the edges outwards. You can use the Show Edges option to view the edge image directly while adjusting this parameter.

blurFromObj: *Default:* .01*width, *Range:* 0 or greater.
Blurs the FromObj input clip before finding the edges. This can help remove noise, and reduce the number of separate shapes. You can use the Show Edges option to view the edge image directly while adjusting this parameter.

show: *Radio buttons, Default: Result.*
Selects the output option.

> ***Result:*** shows the normal lightning result over the background.
> ***Edges:*** shows the edge image. This can be useful to view while adjusting the Threshold and Blur From Obj parameters.

maxLength: *Default:* .1*width, *Range:* 0 or greater.
Scales the length of the bolts.

varyLength: *Default:* 0.5, *Range:* 0 to 1.
The amount to randomly vary the length of each bolt. A value of 0 makes all bolt lengths equal to Max Length, and a value of 1 makes bolt lengths between zero and Max Length.

varySpacing: *Default:* 0.5, *Range:* 0 or greater.
The amount to randomly vary the starting point of each bolt along the edges. A value of 0 makes the bolts regularly spaced, and value of 1 make the bolts randomly spaced.

boltWidth: *Default:* .00625*width, *Range:* 0 or greater.
The width of the lightning bolts.

varyWidth: *Default:* 0, *Range:* 0 to 1.

The amount of random variation in the width of the bolts along their lengths.

endPointiness: *Default:* 1, *Range:* 0 to 1.
Determines how pointed the end of the bolts are. If 0, the entire bolt will have equal width. If 1, the bolts will thin out along their entire length for a pointed end. If it is .5, the bolts will start thinning out half way between the start and end points.

wiggleStart: *Default:* 0, *Range:* 0 or greater.
By default the bolts automatically wiggle over time. This parameter provides a starting offset for these bolt perturbations.

wiggleSpeed: *Default:* 1, *Range:* 0 or greater.
The speed at which the bolts are perturbed automatically over time. To animate changes in speed, set this to zero and animate the Wiggle Start parameter instead.

jitterFrames: *Integer, Default:* 0, *Range:* 0 or greater.
If this is 0, the same random lightning will be used for every frame processed. If it is 1, different random lightning is used for each frame. If it is 2, new random lightning is used for every other frame, and so on.

randSeed: *Default:* 0.123, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different random lightning bolts, and the same value should give a repeatable result.

wrinkleAmp: *Default:* 1, *Range:* 0 or greater.
Scales the amount of wrinkles in the bolts. Decrease for straighter smoother bolts or increase for more kinky bolts.

curveAmp: *Default:* 0.5, *Range:* 0 or greater.
Similar to Wrinkle Amp but affects the general path of the bolt. If decreased, the bolt will stay closer to the line between the Start and End points. If increased it can wander further away from this line. This differs from the Wrinkle Amp parameter in that it can be used to make straighter bolts while still keeping the wrinkles at the detailed level.

branchiness: *Default:* 1, *Range:* 0 to 10.
Scales the number of additional bolts that branch from the main bolt. Set this to 0 for basic bolts with no extra branches.

branchAngle: *Default:* 65, *Range:* 0 to 180.
The maximum angle of the random branches relative to the bolt they are branching off of. If this is 0 the branches will be more lined up with the main bolt. With larger values the branches will be more perpendicular to the main bolt.

branchLength: *Default:* 0.5, *Range:* 0 to 1.
The maximum length of the branches relative to the distance between the Start and End points.

glowColor: *Default rgb:* [0.5 0.5 1].
The color of the glow applied to the lightning.

glowBright: *Default:* 2, *Range:* 0 or greater.
Scales the brightness of the glow applied to the lightning.

glowWidth: *Default:* .025*width, *Range:* 0 or greater.
The width of the glow applied to the lightning.

glowWidthRed: *Default:* 0.5, *Range:* 0 or greater.
The relative red width of the glow.

glowWidthGrn: *Default:* 1, *Range:* 0 or greater.
The relative green width of the glow.

glowWidthBlue: *Default:* 1.5, *Range:* 0 or greater.
The relative blue width of the glow.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the lightning and its glow. The maximum of the red, green, and blue brightness is scaled by this value and combined with the background Alpha at each pixel.

zapColor: *Default rgb:* [1 1 1].
The color of the lightning. If you want to keep the lightning bolt itself bright white, you can still affect the perceived color by adjusting the Glow Color instead.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the lightning bolts.

startOffset: *Default:* 0, *Range:* 0 to 1.
The offset from the start point to begin drawing the bolts. This can be useful for animating a lightning strike.

length: *Default:* 1, *Range:* 0 to 1.
The length of the bolts, beginning at Start Offset. If less than 1, the bolts will not be drawn all the way from start to end. This can be useful for animating a lightning strike.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background before combining with the lightning. If 0, the result will contain only the lightning image over black.

combine: *Radio buttons, Default: Screen.*
Determines how the lightning and glow are combined with the Background.

> ***Screen:*** performs a blend function which can help prevent overly bright results.
> ***Add:*** causes the lightning to be added to the background. This gives brighter glows over light backgrounds.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

blurMask: *Default:* 0, *Range:* 0 or greater.
Blurs the S_Mask input by this amount before using. This can provide a smoother transition between the masked and unmasked areas. This has no effect unless the S_Mask input is connected.

invertMask: *Toggle−button, Default:* off.
If on, inverts the mask input so the effect is applied to areas where the S_Mask is black instead of white. This has no effect unless the S_Mask input is connected.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

Script Form: `S_ZapFrom(FromObject, Background, S_Mask, 1.06, surfaceBolts, etc...);`

See Also:

Zap
ZapTo
Sapphire Plug−ins Introduction

ZapTo

Generates a forked lightning bolt from a given point to the edges of objects in the ToObj input clip, and renders it over a background input.

Inputs:

ToObject: The edges of objects in this clip are extracted, and the lightning connects to points along these edges facing towards the starting point.

Background: *Optional.* The clip to use as background.

Parameters:

surfacePoints: *Integer, Default:* 10, *Range:* 1 to 500.
The number of points along the edges to connect the lightning to. These surface points are divided up amongst the shapes in the ToObj input. If the number of requested surface points is equal to the number of separate shapes in the ToObj input, one lightning fork will connect to each.

start: *X & Y, Default:* [.5*width .5*height], *Range:* any.
The starting position of the lightning.

maxDist: *Default:* width, *Range:* 0 or greater.
The maximum distance of surface points from the Start position. Edges outside this distance are ignored.

threshold: *Default:* 0.5, *Range:* 0 to 1.
The value used to determine the edge locations. Objects darker than this value are ignored. On smooth objects, larger threshold values move the edges inwards to make smaller shapes, and smaller values move the edges outwards. You can use the Show Edges option to view the edge image directly while adjusting this parameter.

blurToObj: *Default:* .003*width, *Range:* 0 or greater.
Blurs the ToObj input clip before finding the edges. This can help remove noise, and reduce the number of separate shapes. You can use the Show Edges option to view the edge image directly while adjusting this parameter.

show: *Radio buttons, Default: Result.*
Selects what the effect will output.

> ***Result:*** shows the normal lightning result over the background.
> ***Edges:*** shows the target edge image. This can be useful to view while adjusting the Threshold and Blur To Obj parameters.

boltWidth: *Default:* .00625*width, *Range:* 0 or greater.
The width of the lightning bolts.

varyWidth: *Default:* 0, *Range:* 0 to 1.
The amount of random variation in the width of the bolts along their lengths.

endPointiness: *Default:* 0.25, *Range:* 0 to 1.
Determines how pointed the end of the bolts are. If 0, the entire bolt will have equal width. If 1, the bolts will thin out along their entire length for a pointed end. If it is .5, the bolts will start thinning out half way between the start and end points.

wiggleStart: *Default:* 0, *Range:* 0 or greater.
By default the bolts automatically wiggle over time. This parameter provides a starting offset for these bolt perturbations.

wiggleSpeed: *Default:* 1, *Range:* 0 or greater.
The speed at which the bolts are perturbed automatically over time. To animate changes in speed, set this to zero and animate the Wiggle Start parameter instead.

jitterFrames: *Integer, Default:* 0, *Range:* 0 or greater.
If this is 0, the same random lightning will be used for every frame processed. If it is 1, different random lightning is used for each frame. If it is 2, new random lightning is used for every other frame, and so on.

randSeed: *Default:* 0.123, *Range:* 0 or greater.
Used to initialize the random number generator. The actual seed value is not significant, but different seeds give different random lightning bolts, and the same value should give a repeatable result.

wrinkleAmp: *Default:* 1, *Range:* 0 or greater.
Scales the amount of wrinkles in the bolts. Decrease for straighter smoother bolts or increase for more kinky bolts.

branchiness: *Default:* 1, *Range:* 0 to 20.
Scales the number of additional bolts that branch from the main bolt. Set this to 0 for basic bolts with no extra branches.

branchAngle: *Default:* 65, *Range:* 0 to 180.
The maximum angle of the random branches relative to the bolt they are branching off of. If this is 0 the branches will be more lined up with the main bolt. With larger values the branches will be more perpendicular to the main bolt.

branchLength: *Default:* 0.5, *Range:* 0 to 1.
The maximum length of the branches relative to the distance between the Start and End points.

bolts: *Integer, Default:* 1, *Range:* 1 to 200.
The number of independent forked lightning bolts to draw, each connecting the Start position with the edge points.

glowColor: *Default rgb:* [0.5 0.5 1].
The color of the glow applied to the lightning.

glowBright: *Default:* 2, *Range:* 0 or greater.
Scales the brightness of the glow applied to the lightning.

glowWidth: *Default:* .025*width, *Range:* 0 or greater.
The width of the glow applied to the lightning.

glowWidthRed: *Default:* 0.5, *Range:* 0 or greater.
The relative red width of the glow.

glowWidthGrn: *Default:* 1, *Range:* 0 or greater.
The relative green width of the glow.

glowWidthBlue: *Default:* 1.5, *Range:* 0 or greater.
The relative blue width of the glow.

affectAlpha: *Default:* 0, *Range:* 0 or greater.
If this value is positive the output Alpha channel will include some opacity from the lightning and its glow. The maximum of the red, green, and blue brightness is scaled by this value and combined with the background Alpha at each pixel.

zapColor: *Default rgb:* [1 1 1].
The color of the lightning. If you want to keep the lightning bolt itself bright white, you can still affect the perceived color by adjusting the Glow Color instead.

brightness: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the lightning bolts.

startOffset: *Default:* 0, *Range:* 0 to 1.
The offset from the start point to begin drawing the bolts. This can be useful for animating a lightning strike.

length: *Default:* 1, *Range:* 0 to 1.
The length of the bolts, beginning at Start Offset. If less than 1, the bolts will not be drawn all the way from start to end. This can be useful for animating a lightning strike.

scaleBackground: *Default:* 1, *Range:* 0 or greater.
Scales the brightness of the background before combining with the lightning. If 0, the result will contain only the lightning image over black.

combine: *Radio buttons, Default: Screen.*
Determines how the lightning and glow are combined with the Background.

> ***Screen:*** performs a blend function which can help prevent overly bright results.
> ***Add:*** causes the lightning to be added to the background. This gives brighter glows over light backgrounds.

pixelAspect: *Default:* GetDefaultAspect(), *Range:* 0.1 to 10.
The aspect ratio of pixels. Higher values squash the pattern vertically, lower values stretch it. This is normally 1.111 for NTSC, .938 for PAL, or 1.0 for square pixels.

clipMode: *Radio buttons, Default:* Union.
Determines how to set the viewport of the output. Select an input name to make the output dimensions the same as that input, or select Union to include all inputs.

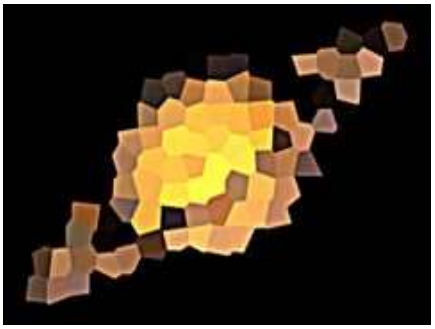
Script Form: `S_ZapTo(ToObject, Background, 1.06, surfacePoints, etc...);`

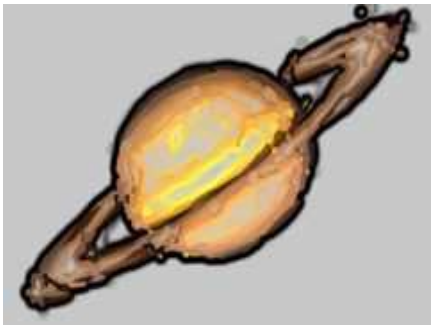
See Also:

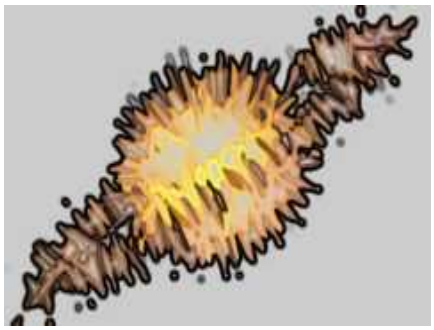
Zap
ZapFrom
Sapphire Plug−ins Introduction

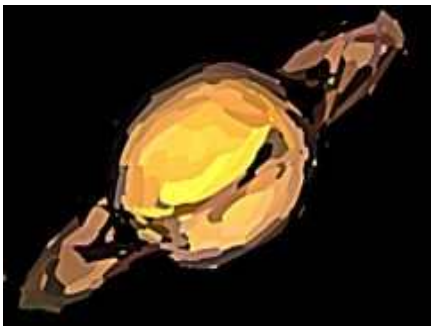
List of Effects With Pictures


AutoPaint:HairyPaint


AutoPaint:Pointalize


AutoPaint:Sketch


AutoPaint:SketchBumpy


AutoPaint:VanGogh


BandPass


BlurMoCurves


BlurMotion


Blur


BlurChannels

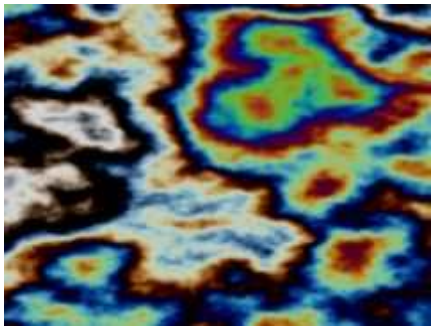

BlurChroma


Clouds

CloudsColorSmooth

CloudsMultColor

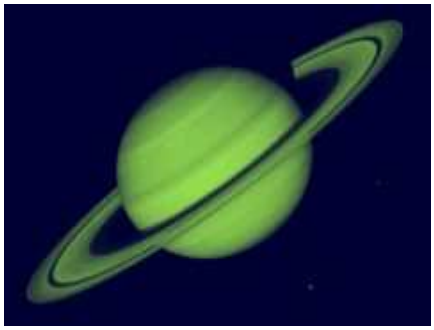
CloudsPerspective



CloudsPsyko

CloudsVortex

ColorOps:ClampChroma

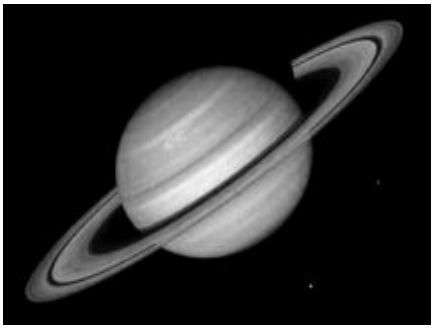


ColorOps:DuoTone

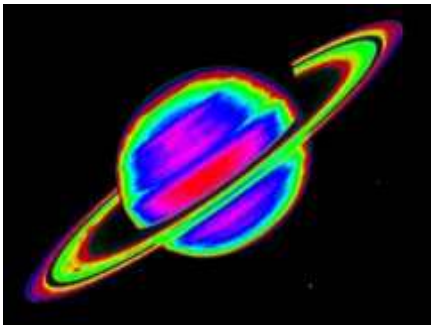
ColorOps:Gamma

ColorOps:Hotspots

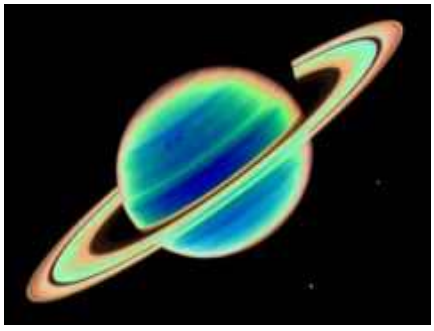
ColorOps:HueSatBright



ColorOps:Monochrome



ColorOps:PseudoColor



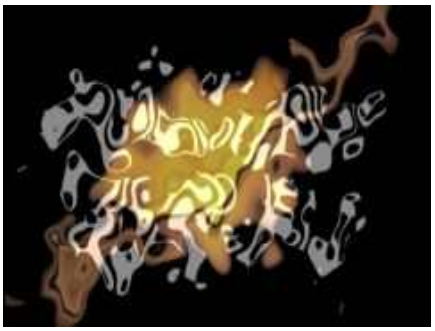
ColorOps:Solarize

ColorOps:Threshold

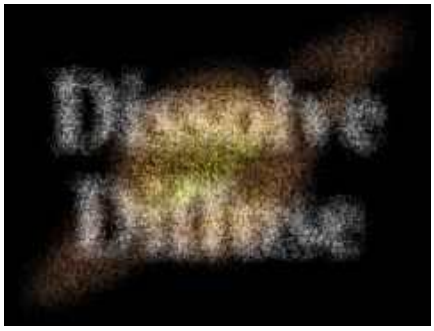
ColorOps:Tint

Diffuse

DissolveBlur



DissolveBubble



DissolveDiffuse

DissolveLuma

DissolvePuddle

DissolveSpeckle

DissolveStatic

DissolveVortex

DissolveWaves

Distort

DistortBlur

DistortChroma

DistortRGB

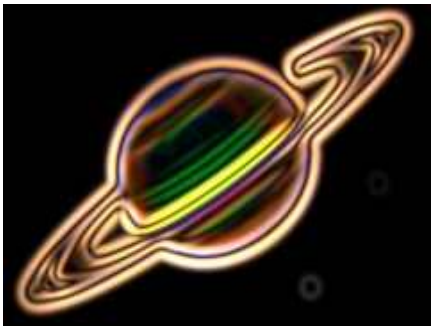
DropShadow

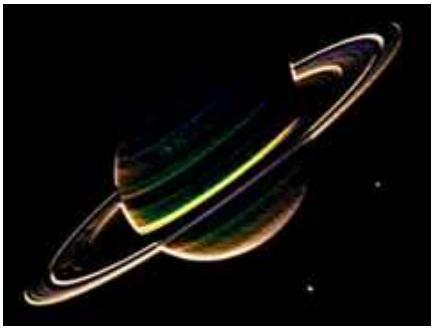
EdgeBlur

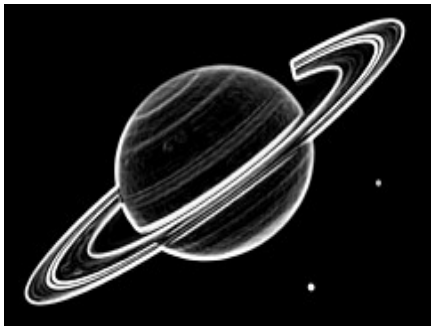

EdgeDetect


EdgeDetectChroma

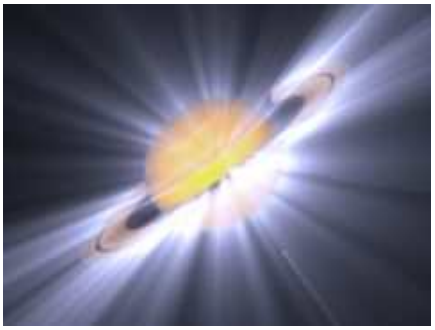
EdgeDetect:Colorize


EdgeDetectDouble


EdgeDetect:InDirection


EdgeDetectMono

EdgeFlash


EdgeRays


Emboss


EmbossDistort


EmbossGlass


EmbossShiny

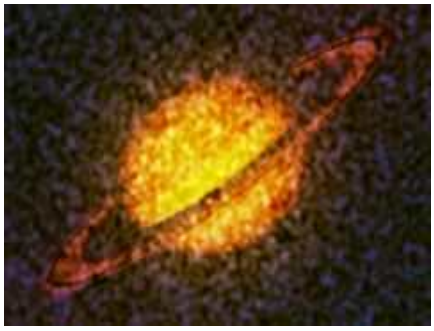
Feedback

FeedbackBubble

Feedback:NearestColor

Feedback:Trails

Feedback:TrailsDiffuse


FilmEffect

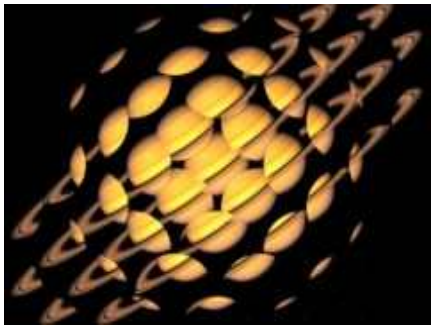
Flicker

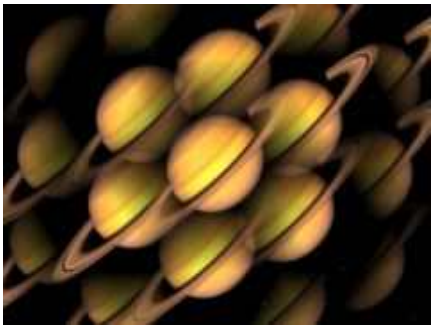
FlickerMatch, Matte

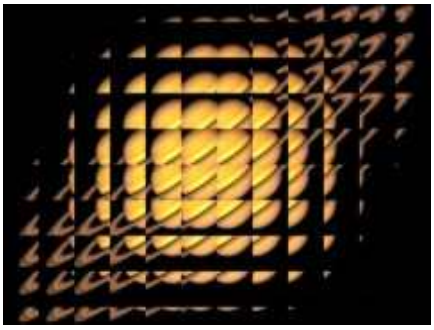
FlickerMatchColor, Matte

FlickerRemove, Matte

FlickerRemoveColor, Matte


FlysEyeCircles


FlysEyeHex


FlysEyeRect

Glare

Glint

GlintRainbow

Glow

GlowAura

GlowDarks

GlowDist

GlowEdges

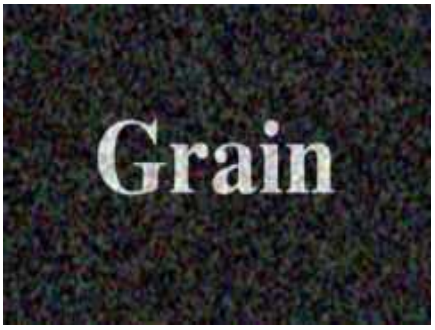
GlowNoise

GlowOrthicon

GlowRainbow

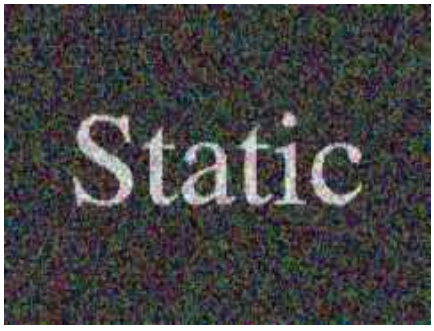
GlowRings


Gradient

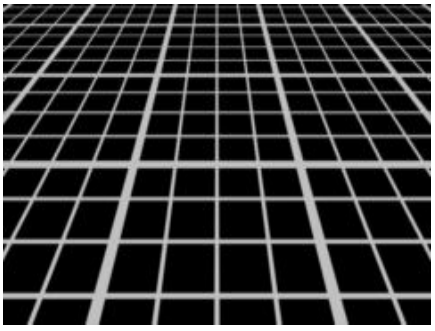


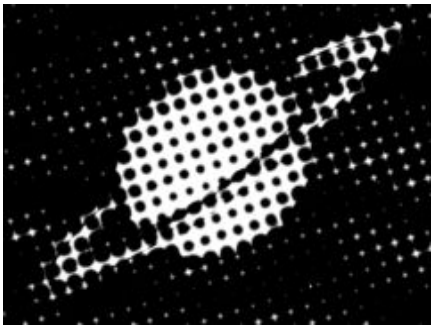
Grain

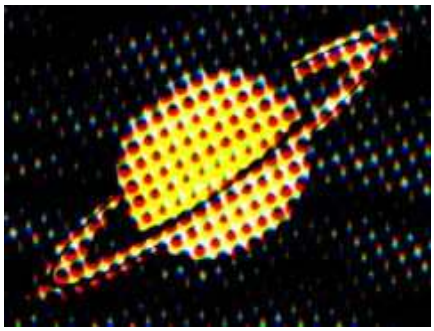
GrainRemove



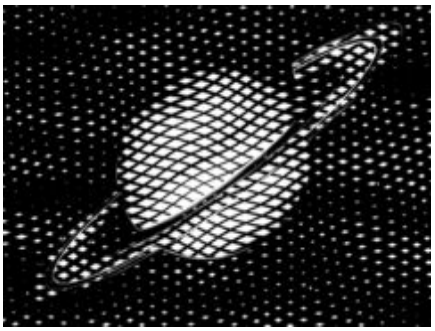
Grain:Static


Grid

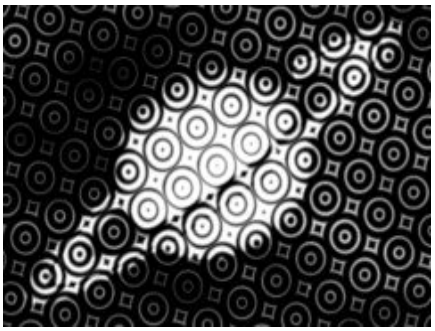

HalfTone



HalfToneColor

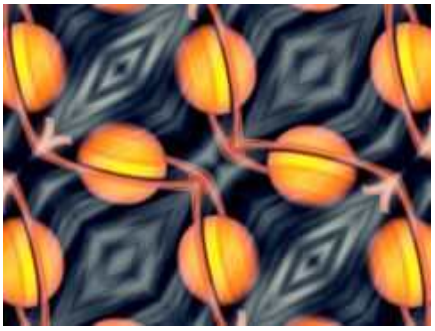


HalfTone:Etching

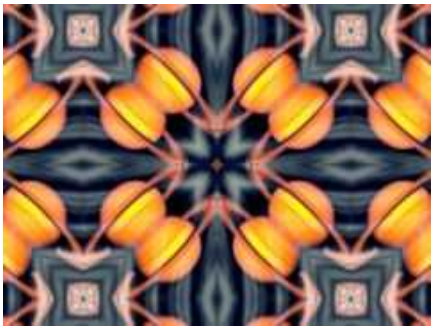


HalfToneRings

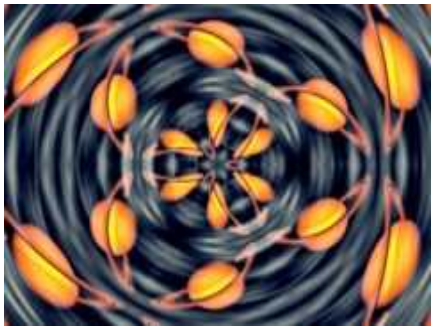
JpegDamage



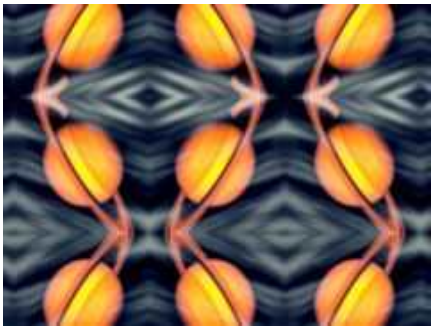
KaleidoDiamonds



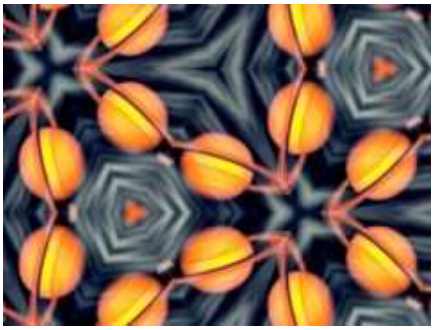
KaleidoOct



KaleidoPolar



KaleidoSquares



KaleidoTriangles

Layer

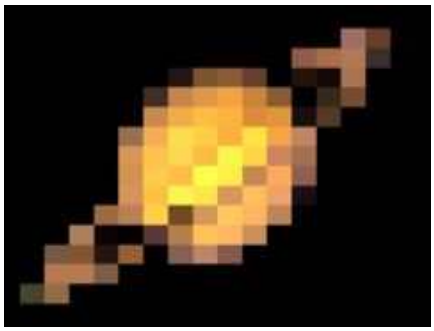
LensFlare

LensFlareAutoTrack

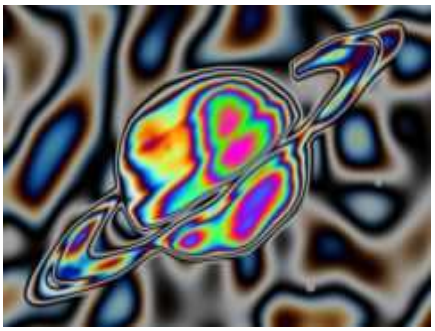
MathOps

MatteOps

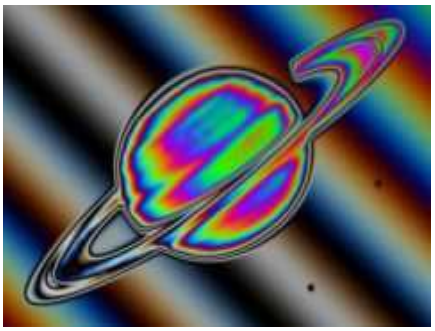
MatteOpsComp



Mosaic



PsykoBlobs

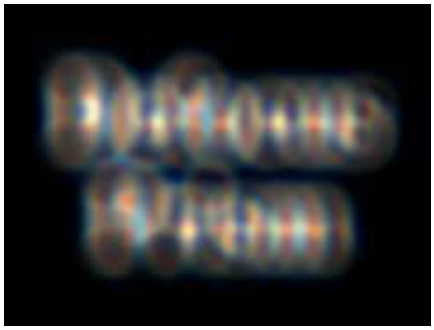


PsykoStripes

Psyko:Zebrafy

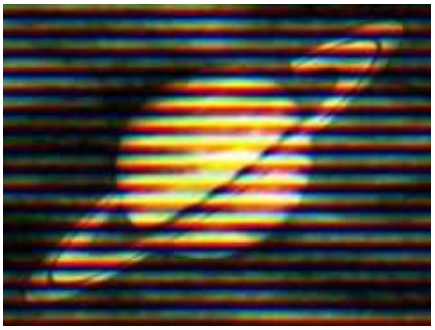
Psyko:ZebrafyColor

RackDefocus

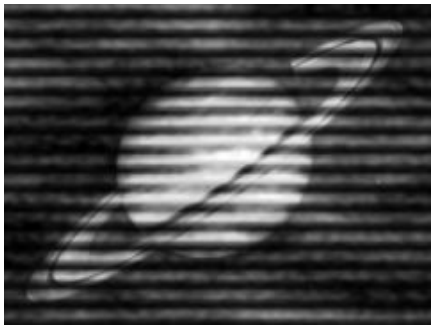


RackDefocus:Prism

RackDfComp



ScanLines



ScanLinesMono

Shake

Sharpen

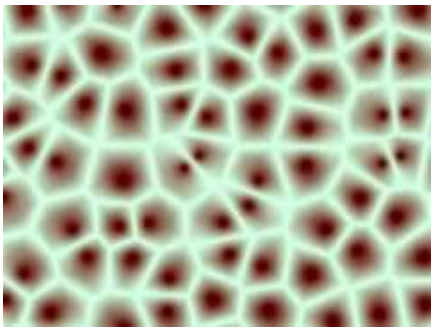
SoftFocus

Sparkles

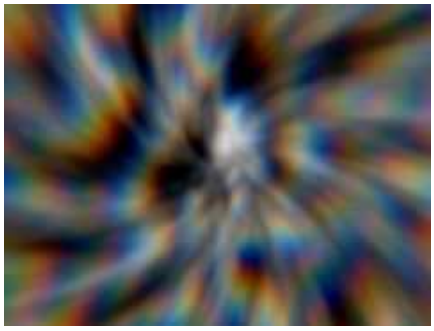
SparklesColor

SpotLight

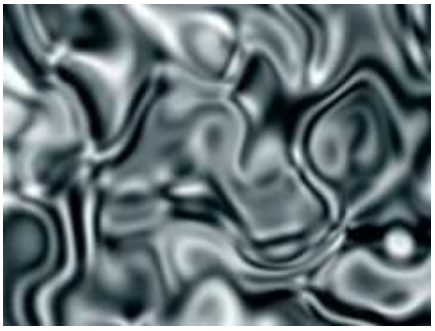
Streaks



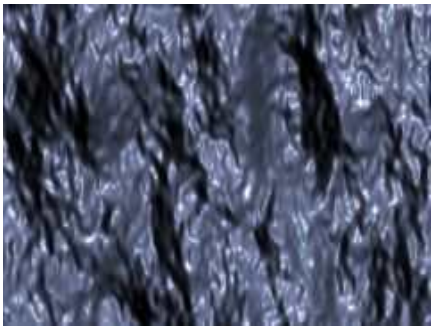
TextureCells



TextureChromaSpiral

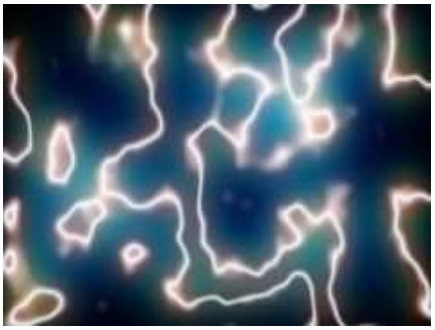


TextureFolded

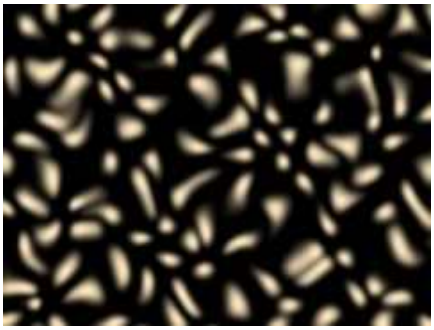


TextureNoiseEmboss

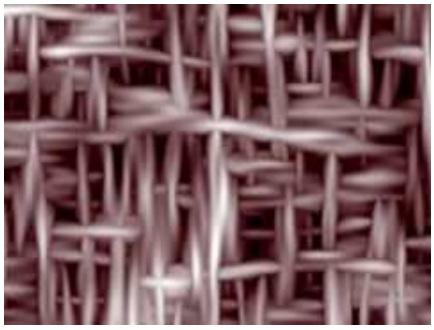
TextureNoisePaint



TexturePlasma

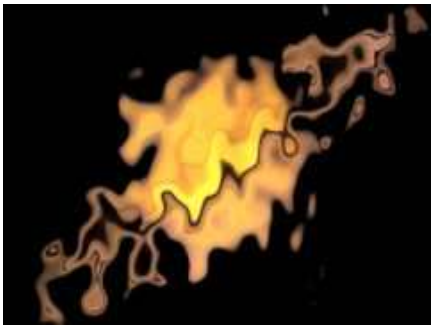


TextureSpots

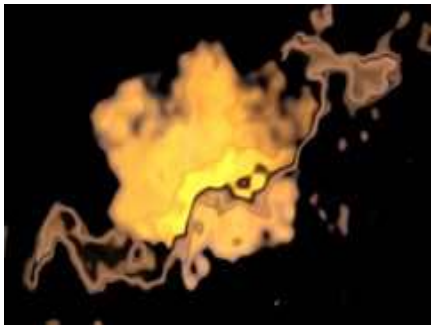


TextureWeave

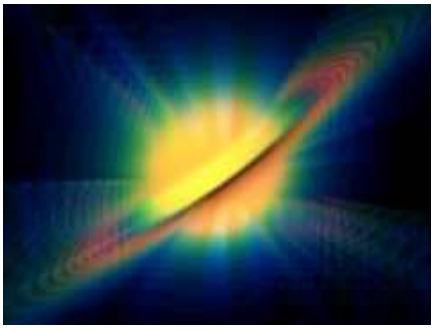
WarpDrops



WarpBubble



WarpBubble2

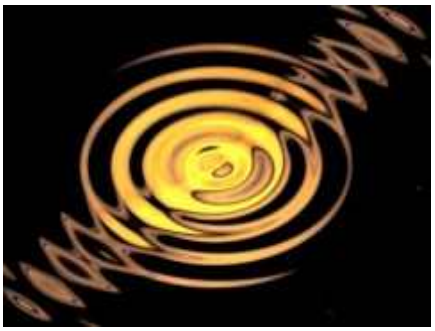


WarpChroma

WarpFishEye

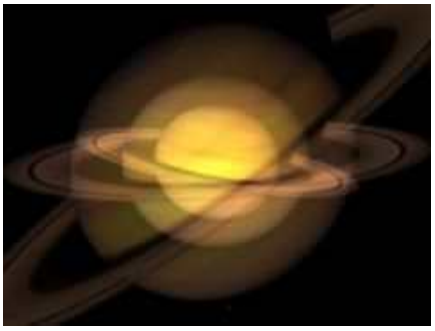
WarpPerspective

WarpPolar



WarpPuddle

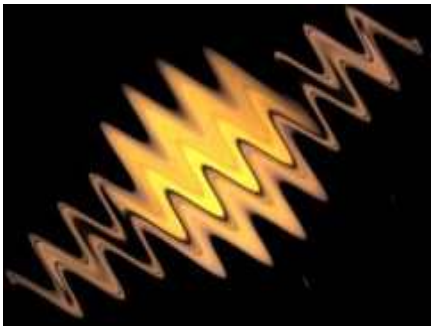
WarpPuff



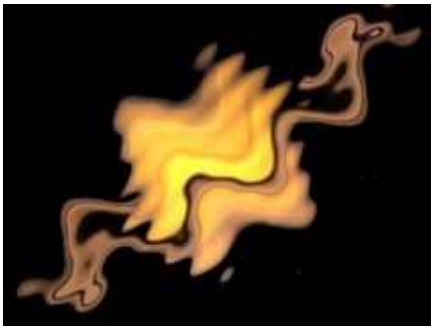
WarpRepeat

WarpTransform

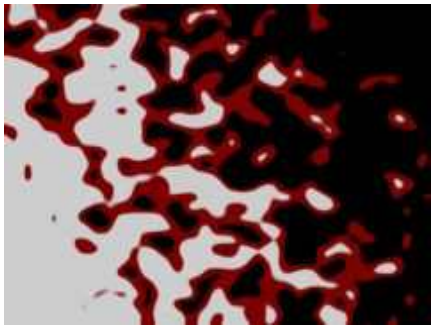
WarpVortex



WarpWaves

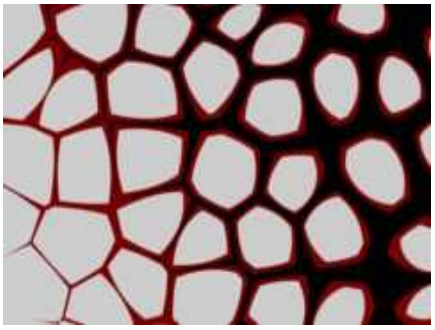


WarpWaves2

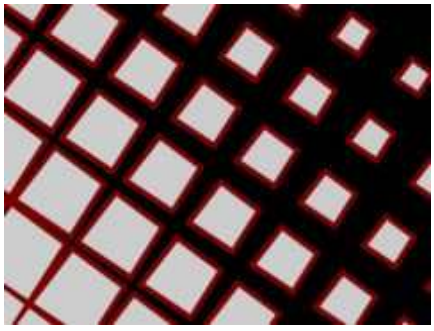


WipeBlobs

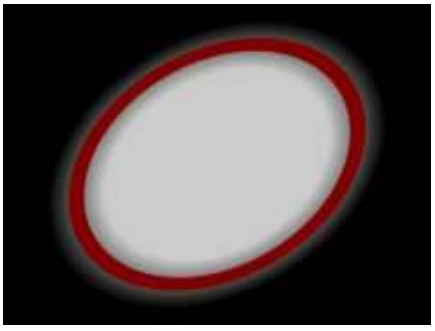
WipeBubble



WipeCells

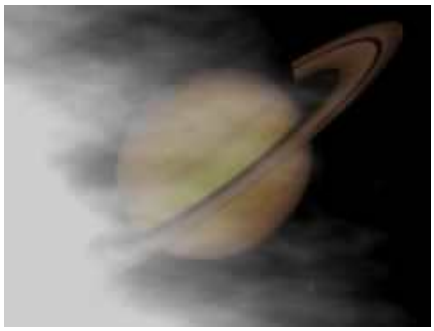


WipeChecker



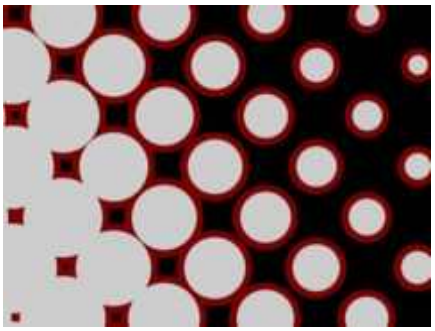
WipeCircle

WipeClock

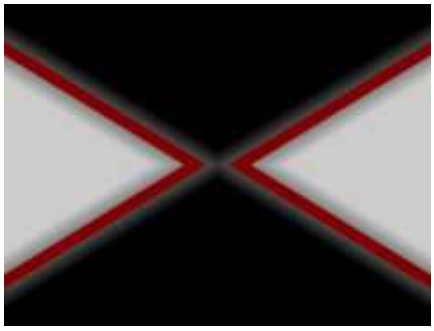


WipeClouds

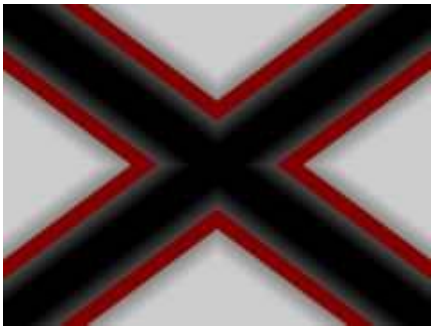
WipeDiffuse



WipeDots

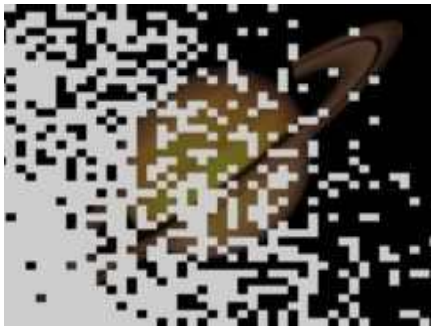


WipeDoubleWedge

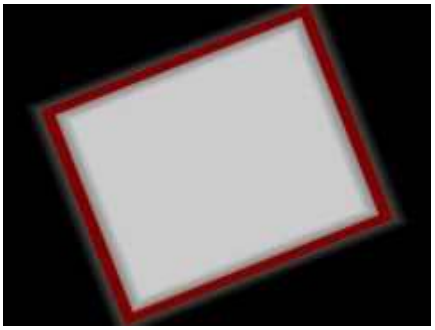


WipeFourWedges

WipeLine



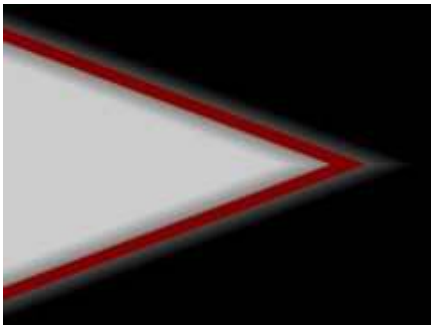
WipePixelate



WipeRectangle

WipeStar

WipeStripes



WipeWedge

ZDepthCueBlur

ZFogExponential

ZFogLinear

Zap

ZapFrom

ZapTo