



***SpeedSix Lenz Raptor V1.2 for Autodesk (Discreet)  
Burn on Linux Red Hat EL WS 3 and 4***

***Red Hat EL WS 3 32 bit:  
Burn 1.5 – 1.7 (32 bit)  
Burn 2.0 or higher (32 bit)***

***Red Hat EL WS 4 64 bit:  
Burn 2.0 or higher (64 bit)***

***14–August–2008***

# Table of Contents

<b>1 SpeedSix Lenz Raptor V1.2 for Autodesk/Discreet Burn on Red Hat Linux</b> .....	<b>1</b>
1.1 Introduction.....	1
1.2 Pre-requisites.....	1
1.3 Downloading the Lenz Raptor.....	1
1.4 Installation.....	2
1.5 Licensing.....	3
1.6 Using a Network License Server.....	4
1.7 Support.....	6
1.8 Sales.....	6
1.9 Thanks!.....	6
<b>2 SpeedSix.CrashZoom (Discreet Lenz Raptor)</b> .....	<b>7</b>
<b>3 SpeedSix.Flare (Discreet Lenz Raptor)</b> .....	<b>8</b>
<b>4 SpeedSix.LensCorrect (Discreet Lenz Raptor)</b> .....	<b>11</b>
<b>5 SpeedSix.LensFlare (Discreet Lenz Raptor)</b> .....	<b>29</b>
<b>6 SpeedSix.RackDefocus (Discreet Lenz Raptor)</b> .....	<b>47</b>

# 1 SpeedSix Lenz Raptor V1.2 for Autodesk/Discreet Burn on Red Hat Linux

## 1.1 Introduction

This manual describes the SpeedSix Lenz Raptor for Autodesk/Discreet "Advanced Systems" on Burn. Lenz is a set of five plugins that are concerned with various aspects of real cameras and lenses.

- ◆ **LensCorrect:** Removes and inserts curvature due to lens distortion.
- ◆ **LensFlare:** Simulates a wide variety of lens flare effects.
- ◆ **Flare:** Lens flares, optionally around multiple light sources.
- ◆ **CrashZoom:** Simulates focal length changes or dollying while the camera shutter is open.
- ◆ **RackDefocus:** Realistic out-of-focus effect with aperture shapes (or any other point spread function).

Below we tell you how to install the Lenz Raptor and how to license it.

Please note that the Lenz Raptor is fully functional without a license – a SpeedSix logo will be superimposed on all output frames, however.

## 1.2 Pre-requisites

Please ensure that your Autodesk/Discreet software is an appropriate version. The 32 bit version of the Lenz Raptor is intended for use with these 32 bit programs:

- Burn V1.5 – V1.7
- Burn V2.0 (32 bit)

The following Autodesk/Discreet products are 64 bit programs and need the 64 bit version of the Lenz Raptor:

- Burn V2.0 and higher. (64 bit)

## 1.3 Downloading the Lenz Raptor

Download the latest version of the Lenz Raptor from:  
<ftp://ftp.speedsix.com/Raptors/discreet/Lenz>

## 1 SpeedSix Lenz Raptor V1.2 for Autodesk/Discreet Burn on Red Hat Linux

You will find separate installation packages for IRIX 32 bit and 64 bit, Linux 32 bit and 64 bit, and Burn 32 bit and 64 bit (six packages in total). Choose the Burn package you need for compatibility with your Autodesk/Discreet product.

The package names are of the form:

Lenz{32|64}\_V1.2.{release}\_burn.tar.gz

E.g.

Lenz64\_V1.2.701\_burn.tar.gz

## 1.4 Installation

To install the Lenz Raptor proceed as follows.

1. Obtain the appropriate installer package (see above). You will usually obtain this from the SpeedSix FTP site, but it may occasionally be provided on CD by your reseller.
2. Create a temporary directory in which to unpack the distribution. E.g.

```
your_prompt> mkdir /var/tmp/s6
```

3. Copy the distribution file from its current location to the temporary directory. E.g.

```
your_prompt> cp /somedisk/mydownloads/Lenz64_V1.2.701_burn.tar.gz /var/tmp/s6
```

4. Go to the temporary directory:

```
your_prompt> cd /var/tmp/s6
```

5. Unpack the distribution file (replace 64 with 32 if on a 32 bit system):

```
your_prompt> tar xzvf Lenz64_V1.2.701_burn.tar.gz
```

6. Become super-user if you have not already done so:

```
your_prompt> su  
Password: <enter your root password>
```

7. Start the installation procedure (omit `-64` in the `cd` if on a 32 bit system):

```
root_prompt> cd discreet-burn-lenz-raptor-64-1.2-nnn-dist (e.g. nnn=701)
root_prompt> ./install_s6
```

8. You will be asked if you are ready to read the SpeedSix License Agreement. Respond **y[enter]** or **n[enter]**. Installation will end at this point if you choose **n[enter]**.

9. After reading the SpeedSix License Agreement you will be asked if you accept it or not. Choose **y[enter]** or **n[enter]**. Installation will end at this point if you choose **n[enter]**.

10. Everything needed will then be automatically installed. (There are no options to worry about).

11. Finally, the installer checks if you have a valid license for each box and bundle, and copies an appropriate proxy image. This will appear when browsing for Sparks in proxy mode in your Discreet application and lets you easily tell which plugins you have licenses for.

### 1.4.1 After Installation

When the Lenz Raptor has been installed, you will have the following:

- **/usr/discreet/sparks/SpeedSix\_V1.2\_Raptors** or **SpeedSix\_V1.2\_Raptors64** is where you will find the new Lenz Raptor Sparks.
- **/usr/local/SpeedSix/Licenses** is where the Lenz Raptor license lives. When you get a license file from SpeedSix, it should be copied **unaltered** to this directory. See below for more information. Also see below for information on using the Network License Server as an alternative to node locked licenses.
- **/usr/local/SpeedSix/bin** contains SpeedSix utility programs – mainly for licensing. See below for more information.
- **/usr/local/SpeedSix/Docs** contains a PDF format manual for the Lenz Raptor (derived from the HTML help), which you can view with (for example) **xpdf** or print. The manual is located at:  
**/usr/local/SpeedSix/Docs/Discreet\_Burn\_Lenz\_Raptor\_PDFManual.pdf**

## 1.5 Licensing

You do not need a license key to run the software for evaluation purposes, but the images it creates will be watermarked.

To remove the watermarks, you will need a SpeedSix license file. This will be provided when you purchase the software.

## 1 SpeedSix Lenz Raptor V1.2 for Autodesk/Discreet Burn on Red Hat Linux

You can choose to use node locked licensing or to use "floating" licenses supplied from the SpeedSix Network License Server.

We first describe how to use a node locked license.

Your Lenz Raptor license is tied to the MAC address of the **eth0** Ethernet adaptor of your machine. The easiest way of displaying this number in the right format is to use:

```
your_prompt> /usr/local/SpeedSix/bin/ssid
```

after you have installed the Lenz Raptor.

Alternatively, you can use this Linux command:

```
your_prompt> /sbin/ifconfig eth0
```

and send us the **HWaddr: xx:yy:zz:aa:bb:cc** part of the output.

SpeedSix (or your reseller) will send you your license as an email attachment. Proceed as follows:

1. Save the attachment as a file. This *is* the license – ***please keep a copy in a safe place.***
2. Copy the file to the following directory on the machine to be licensed:

```
/usr/local/SpeedSix/Licenses
```

Your Lenz Raptor is now licensed and will render without watermarks next time you start the Discreet application and load a Lenz Raptor plugin.

### PLEASE NOTE

Do not edit the license files provided in any way. The licenser searches all the files in the License directory to find valid licenses, so more than one license file can be added to the directory without difficulties arising.

## 1.6 Using a Network License Server

You can choose to install the SpeedSix License Server on one of your Linux machines if you wish. This allows "floating" licenses so that you can swap out **burn\*** rendering machines without worrying about licensing problems (amongst other things).

First, download the License Server from:

[ftp://ftp.speedsix.com/license\\_server/sslserver-linux-v1.2-702.tar.gz](ftp://ftp.speedsix.com/license_server/sslserver-linux-v1.2-702.tar.gz)

or, if you will be running the License Server on a 64 bit machine:

[ftp://ftp.speedsix.com/license\\_server/sslserver-linux-64-v1.2-702.tar.gz](ftp://ftp.speedsix.com/license_server/sslserver-linux-64-v1.2-702.tar.gz)

## 1 SpeedSix Lenz Raptor V1.2 for Autodesk/Discreet Burn on Red Hat Linux

To install the License Server, and for a full description of how to administer licenses with the License Server, please consult the document:

[ftp://ftp.speedsix.com/license\\_server/SslServerLinux.pdf](ftp://ftp.speedsix.com/license_server/SslServerLinux.pdf)

To use network licenses, you must tell your licensing machines to look for licenses on the license server. You do this with the **ssladmin** program. Assuming that the SpeedSix binary directory (/usr/local/SpeedSix/bin on Linux) is in your path, use:

```
prompt> ssladmin

SpeedSix Licensing - Administration Program V1.2
=====

ssladmin> set server 192.168.1.10
ssladmin> exit
```

You need to do this on each of your rendering nodes and on your license server node.

Licenses are then issued for the machine running the license server. To get the system id to send to SpeedSix when requesting licenses use the **ssid** program on the license server machine:

```
prompt> ssid

SpeedSix Licensing - System Identification Number
=====

System id = 00e0810095ec
```

### NOTE

**ssladmin** and **ssid** are installed with both the Lenz Raptor and the License Server installation so they are available on both the client and the server ends.

When you get a license, it will be delivered electronically as an e-mail attachment. Copy it to the SpeedSix licenses directory on the license server machine. This is:

/usr/local/SpeedSix/Licenses

You can tell the license server that new licenses have been added using **ssladmin**:

```
prompt> ssladmin

SpeedSix Licensing - Administration Program V1.2
=====

ssladmin> update
Updated knowledge of local licenses.
Updated license database on server: 192.168.1.10

ssladmin> exit
```

Note that you can perform almost all license administration tasks using **ssladmin** from any machine it is installed on since **ssladmin** communicates with **sslserver** when required (the server location must be set, of course).

There are many other **ssladmin** functions – using `?` or `help` commands in **ssladmin** gives a brief description of all of them.

Complete documentation can also be found in **SslServerLinux.pdf**. You can view this with **xpdf** on Linux (if **xpdf** is installed).

## 1.7 Support

In the event of any difficulties with the installation or the software please contact [support@speedsix.com](mailto:support@speedsix.com)

## 1.8 Sales

For sales enquiries contact [sales@speedsix.com](mailto:sales@speedsix.com), or see [www.speedsix.com](http://www.speedsix.com). You will need to contact your reseller or SpeedSix sales to obtain a license.

## 1.9 Thanks!

Thanks for using SpeedSix software!

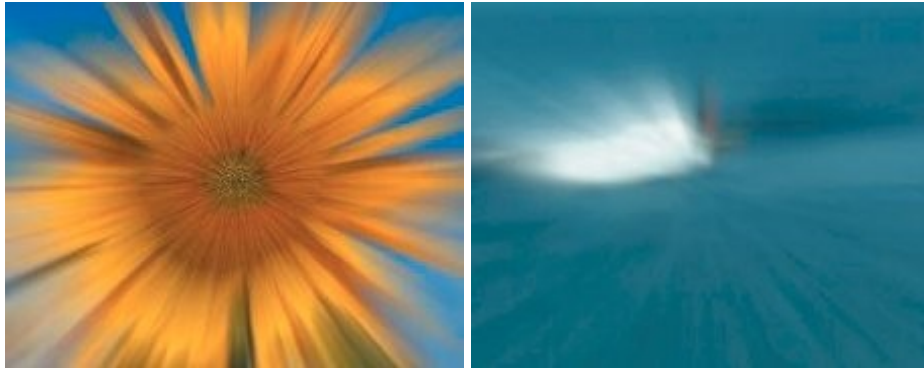


## 2 SpeedSix.CrashZoom (Discreet Lenz Raptor)

---

### PURPOSE

Simulates the effect of holding the shutter open in a camera with a zoom lens while swiftly changing the focal length of the lens. Similar effects would happen if a camera were dollied very quickly towards a target. Might also be called convergence blur.



### INPUT CLIPS

1: Clip to Zoom

---

### *CrashZoom* CONTROL PAGE

**Centre** (Position **Default:** 0.5,0.5)

The convergence position on the image. The co-ordinates need not lie within the image.

**Clear Radius** (Number **Min:** 0.0, **Max:** 10.0, **Default:** 0.4)

The radius of a circular region centred on the 'Centre' position within which the image is unaffected.

**Speed** (Number **Min:** -200.0, **Max:** 200.0, **Default:** 10.0)

Moving out radially from the centre, every pixel outside the clear centre region moves over a distance that is proportional to its distance from the edge of the clear region during the 'exposure time' of the 'film'. 0.0 – no movement; > 0.0 – zoom towards; < 0.0 – zoom away.

**Sample** (List Box **Options:** 1 Sample | 4 Samples | 16 Samples | 64 Samples, **Default:** 1 Sample)

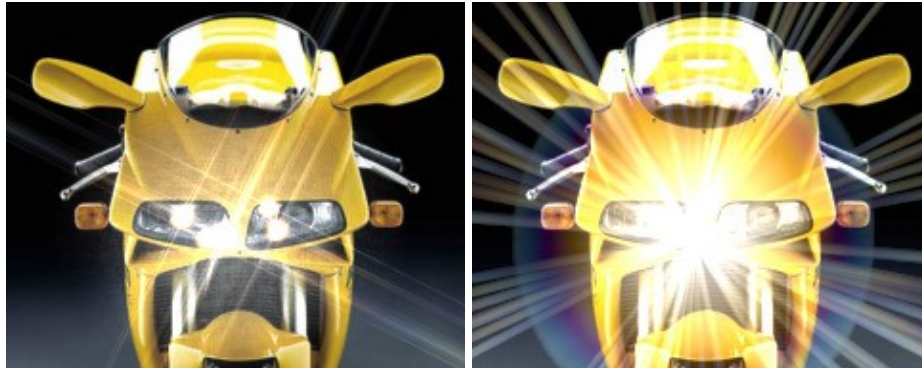
With high contrast images you may see unwanted stepping. Increase the Sample rate to smooth the image. However the higher the rate, the slower the rendering.

## 3 SpeedSix.Flare (Discreet Lenz Raptor)

---

### PURPOSE

Uses a variety of tricks for mimicking the appearance of bright lights in a scene viewed by eye or by a camera. Ideally it requires as input a light map which has pixels set to the colour of lights (i.e. it contains only the light emitting surfaces in a scene).



### INPUT CLIPS

**1: Light Map** : Using this clip flares will be created for **every** pixel at the chosen intensity level. Select judiciously.

**2: Background** : The flares will be drawn on this clip.

---

### ROI CONTROL PAGE

**Light Map Only** (Checkbox **Default: On**)

**On:** When starting to use this effect Light Map Only is always **On**. This displays pixels within the bounding box and within the Low and High range you can see very quickly which pixels will be used for the flare generation.

**Off:** Use when you have found the pixels you want for the full flare effect. Be prepared for a wait if your chosen area is densely packed!

**Match Scales** (Checkbox **Default: On**)

Match Scales

**Scale X** (Number **Min:** 0.01, **Max:** 2.00, **Default:** 0.20)

**Centre** (Position **Default:** 0.2,0.3)

**Scale Y** (Number **Min:** 0.01, **Max:** 2.00, **Default:** 0.20)

Delimits the bounding box. Only pixels within the box will be considered for flares.

**Low** (Colour Box **Default: R: 58982, G: 58982, B: 58982, A: 0**)

**Low Light** (Number **Min: 0.000, Max: 100.000, Default: 90.0**)

**High Light** (Number **Min: 0.000, Max: 100.000, Default: 100.000**)

**High** (Colour Box **Default: R: 65535, G: 65535, B: 65535, A: 0**)

Use to bound the intensity levels to consider when creating the flares.

---

## **Flares CONTROL PAGE**

**Aperture (mm)** (Number **Min: 0.1, Max: 20.1, Default: 4.0**)

Controls the diameter of lens or eye pupil aperture. The smaller this is, the wider the 'rainbow' halo diffraction effect will be.

**Halo Fringes** (Number **Min: 0, Max: 100, Default: 100**)

This controls the number of radial lines eating in to the central halo. It is combined with random weightings to give the final effect. In the eye, the central brightest part of a light source image often looks as if it has been 'eaten into' by radial lines, so it's no longer symmetrical and smooth (even when the original light source is). This is due mainly to inconsistencies in the material of the eye. It doesn't happen with artificial lenses.

**HaloStrength** (Number **Min: 0.1, Max: 100.0, Default: 50.0**)

Adjusts the relative importance of the halo effect.

**Halo FrnStr** (Number **Min: 0.0, Max: 100.0, Default: 50.0**)

Controls how much (on average) each radial line halo fringe eats in to the central halo.

**Contrast** (Number **Min: 0.50, Max: 300.00, Default: 135.0**)

Controls the overall contrast of the combined flare effects before they are composited over the background image. Can usually be left at the default, or perhaps lowered to make the effect more subtle.

**Streaks** (Number **Min: 0, Max: 1000, Default: 2**)

Specifies the number of streaks to use. These are centred on the light source and are somewhat random in orientation and length (but consistent from frame to frame throughout a sequence!).

**Mono Streak** (Number **Min: 0.0, Max: 100.00, Default: 5.0**)

Increases or reduces the colour content of the streaks.

**Streak Length** (Number **Min: 0, Max: 1000, Default: 75**)

Sets the maximum streak length. They can be up to 50% shorter.

**Streak Rot** (Number **Min: 0.0, Max: 360.0, Default: 0.0**)

Animating this will give some of the effects seen in streaks when the observer moves.

**Streak Strength** (Number **Min:** 0.1, **Max:** 100.0, **Default:** 10.0)

Adjusts the relative importance of the streak effect.

**Glow Radius** (Number **Min:** 1.0, **Max:** 200.0, **Default:** 10.0)

Controls the radius of the glow effect around each pixel in the light map. These separate glow effects add up to form the overall glow around an extended light source.

**Glow Strength** (Number **Min:** 0.1, **Max:** 100.0, **Default:** 40.0)

Adjusts the relative importance of the glow effect.

**Reflections** (Number **Min:** 0, **Max:** 10, **Default:** 0)

Controls how many times a light source is reflected. In reality, this depends on the number of surfaces (e.g. lens elements) there are in the path of the light.

**Reflectivity** (Number **Min:** 0.0, **Max:** 100.0, **Default:** 20.0)

How much of the light source intensity is reflected at each reflecting surface.

**Refl Scale** (Number **Min:** 10.0, **Max:** 1000.0, **Default:** 100.0)

How reflected positions of lights are scaled about the center of the image. The reflecting surfaces are treated as a set of evenly spaced parallel plates. Animating this control can add a bit more interest.

## 4 SpeedSix.LensCorrect (Discreet Lenz Raptor)

---

### PURPOSE

Corrects for (or adds) lens distortion. It uses a model of lens distortion that includes symmetric (radial) and asymmetric (x and y curvature) distortions. **LensCorrect** can semi-automatically remove distortions by estimating the radial distortion parameters that will best make what should be straight lines in the input image into actually straight lines in the output image. This works by having the user mark points along one or more lines in the input that should be straight (e.g. the edges of a building). Only symmetric distortions can be corrected for using this technique – there are too many degrees of freedom when asymmetric distortions are also present. In many cases, however, this "line straightening" method works very well. (Anamorphic – "squeeze" – lenses would be an example of a lens type where there might be significant asymmetric distortion, and hence an example of where "line straightening" would not be sufficient to correct for the distortions present.)



### INPUT CLIPS

1: Clip to process

---

### *LensCorrect* CONTROL PAGE

**Radial1** (Number Min: -1.0, Max: 3.0, Default: 0.0)

Primary radial distortion factor. This accounts for the majority of the symmetric distortion.

**Radial2** (Number Min: -1.0, Max: 3.0, Default: 0.0)

Secondary radial distortion factor. This accounts for "extra" radial distortion near the lens edges.

**XCurvature** (Number Min: -1.0, Max: 1.0, Default: 0.0)

Asymmetric distortion proportional to distance from the center along the X axis only. Usually significant only with lenses that squeeze the image along one axis.

**YCurvature** (Number **Min:** -1.0, **Max:** 1.0, **Default:** 0.0)

Asymmetric distortion proportional to distance from the center along the Y axis only. Usually significant only with lenses that squeeze the image along one axis.

**Centre** (Position **Default:** 0.5,0.5)

Position of the centre of the lens. Usually, this will be at the centre of the image.

**Squeeze** (Number **Min:** 0.5, **Max:** 5.0, **Default:** 1.0)

Horizontal squeeze factor for the lens. Some lenses compress the image along one axis (X axis, usually) for use with some wide screen film formats.

**Sampling** (List Box **Options:** Nearest | Linear | Cubic | SharpCubic | BSpline, **Default:** SharpCubic)

Interpolation type to use when resampling the image.

**Magnify** (Number **Min:** 0.2, **Max:** 5.0, **Default:** 1.0)

Magnification for the output image. This may well be required to avoid black at the edges when correcting for lens distortion.

**LineEdit** (List Box **Options:** No Action | Add Line | Delete Line | Move Point | Delete Point | Insert Point, **Default:** No Action)

Choose an editing action for creating or modifying the lines needed for automatic parameter estimation. You should draw a line along one or more features in the input image (e.g. while having View set to Input) that should be straight.

**KillLines** (Pushbutton)

Delete all the lines you have drawn for automatic parameter estimation.

**AutoFind** (Pushbutton)

Attempt to automatically set **Radial 1** and **Radial 2** from user supplied line data.

**Mode** (List Box **Options:** Remove | Add | Auto Remove, **Default:** Remove)

Apply distortion, remove distortion, or automatically remove distortion. Auto Remove will be selected automatically when you use the Auto button.

**View** (List Box **Options:** Result | Input, **Default:** Result)

Output the result image, or simply copy the input image. You need to view the input image using this control (set to **Input**) when editing straight lines for automatic radial distortion removal.

**Reset** (Pushbutton)

Reset all controls to default values.

---

### **Semi-automatic Distortion Removal by Line Straightening**

If there are straight line features in your input image – ideally with some of them near the image edges – and you don't think that there are likely to be significant asymmetric distortion components present (which will usually be true unless anamorphic lenses are involved), you can have **LensCorrect** estimate the radial distortion parameters that will correct for the distortion in your input image.

**Aside:** Note that **LensCorrect** is **adding a distortion** here that will "undo" the distortion it estimates is in your image. The fact that **LensCorrect** considers it is **adding** a distortion is irrelevant **unless** you subsequently wish to **add back** the original distortion. In that case, you need to use another **LensCorrect** set to **Remove Mode** – and with the **Radial 1** and **Radial 2** parameters set to those found by the **LensCorrect** that has done the "automatic removal". This point will be discussed further in the section on reapplying a lens distortion.

To use semi-automatic distortion removal, proceed as follows:

- 1. Start the **LensCorrect** plugin with the distorted image sequence as its input. An example of a distorted image is shown here:



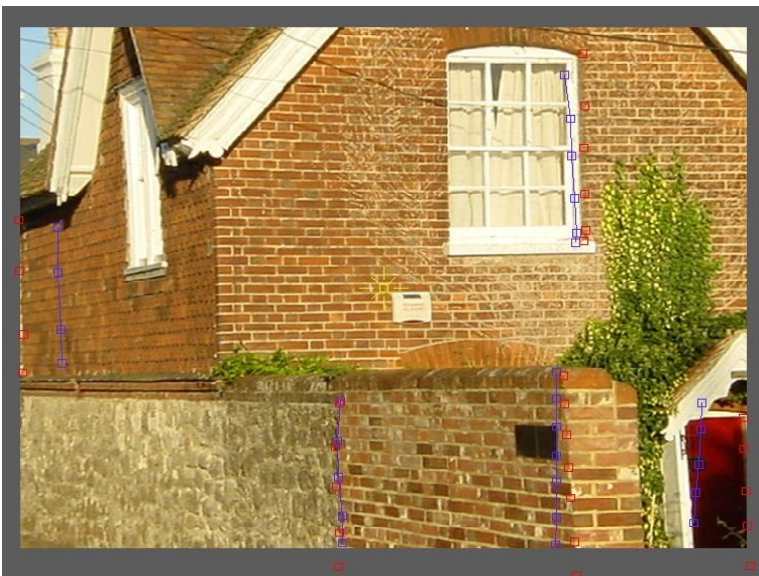
- 2. If you have not just started or added a new **LensCorrect**, press **Reset** to re-initialize the plugin.
- 3. Select **View: Input** so that you are looking at the distorted input image.
- 4. Select **Mode: Auto Remove**
- 5. Pick out an image feature that should be a straight line (in an undistorted image).
- 6. Select **LineEdit: Add Line**. Move the cross hair cursor to the point at which you wish to start the line and press the left mouse button or press the pen down if you are using a digitizing tablet.
- 7. Add more points to the line by moving the cross hair along the image feature you are tracing out and pressing down the pen/mouse. You don't need to add very many points usually. A minimum of 4 and a maximum of 10 might be a good guideline.



- 8. End the line by selecting: **LineEdit: No Action**. Shown below is an example of a set of lines being entered:



- 9. Repeat Step 5 to 8 to create several lines. You won't need too many lines. A guideline would be 2 to 10 lines in total. Ideally, some (or all if there are only a few lines) would be near the edge of the image where the distortion will be most pronounced. With a radial distortion model, there will be no distortion in the center of the image, so lines near the middle won't contribute much information about the distortion that is present.
- 10. When you think you have enough lines, press: **Auto Find. LensCorrect** will then attempt to find a distortion that straightens out the lines you have digitised in. It will set the **Radial 1** and **Radial 2** parameters to the distortion it has calculated will straighten the lines. When it has finished, it will automatically set: **View: Result** so that you can see the results of its efforts. It will also show you where each point you digitised in has been moved to by the distortion (these points are shown a small red boxes):





- 11. If the straight line features in the result now actually appear to be straight, and you are generally satisfied with the result, you may be done. Assuming that the distortion is constant for the length of the input image sequence, you can simply run the sequence through **LensCorrect** without changing anything else. Here is an example of a corrected image:



**Note** that if the lens is a **zoom lens** and the **zoom setting changes** over the sequence, it is **very likely that the distortion will also change!** In that case, you will need to follow the **Auto Remove** process described above at several points in time along the sequence and make notes of the calculated **Radial 1** and **Radial2** parameters at those points. You will then need to set **LensCorrect** to **Add Mode** and enter those radial distortion factors at the appropriate frames, creating a **key framed** sequence of radial distortion parameters. Gross changes in focus settings may also change the distortion characteristics of a lens.

You can adjust the lines you have drawn using the other **LineEdit** options:

- **Delete Line:** Selecting any point on a line will delete the entire line.
- **Move Point:** Selecting a point on a line will let you move that point by dragging the pen/mouse (press down the pen/mouse button, move whilst down, release the pen/mouse button to stop dragging the point).
- **Delete Point:** Selecting a point on a line will remove that point from the line.
- **Insert Point:** Selecting a point and holding the pen/mouse button down will add a new point to the line after the one you selected and let you position it as you wish.

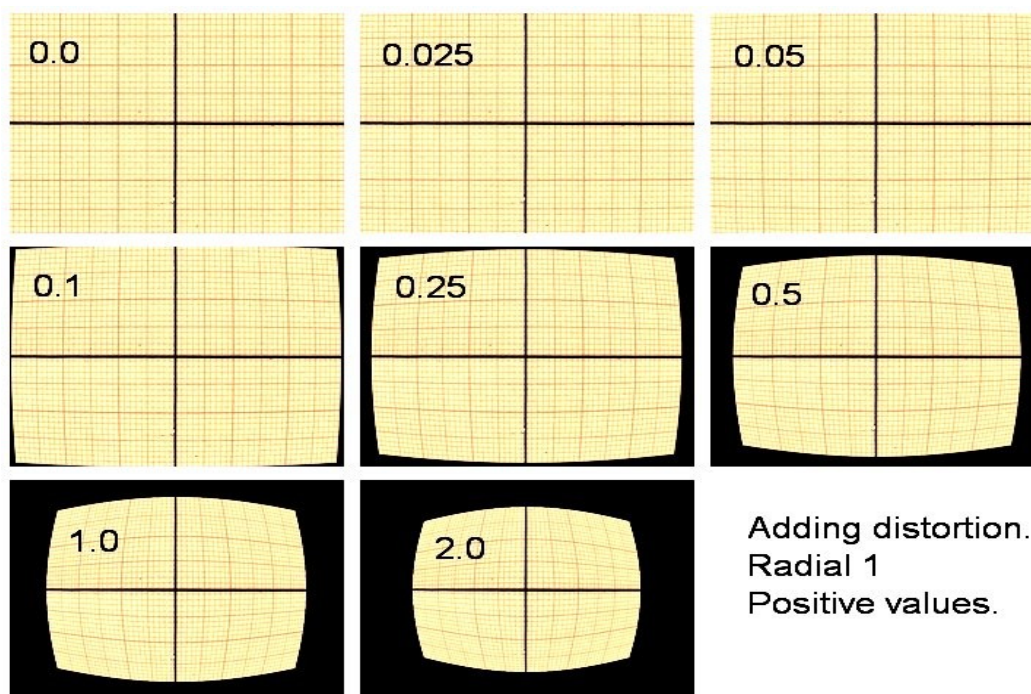
---

### **Manual Distortion Addition and Removal**

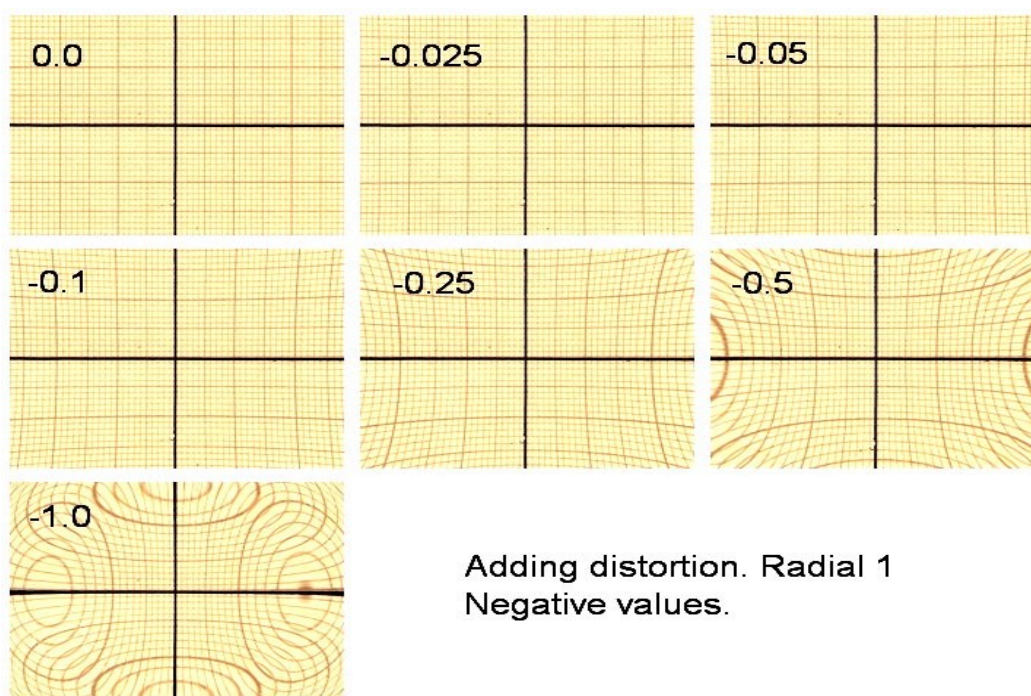
You can try to correct for lens distortion manually simply by adjusting the distortion parameters and watching what happens to the image (look particularly at features you think should be straight lines).

The dominant parameter is **Radial1**. This controls a radially symmetric distortion that gets more and more pronounced as you move away from the lens center (usually the middle of the image) in any direction. In fact, this distortion is proportional to the distance from the lens center squared. Lenses in which the magnification **decreases** as you move away from the lens center towards the edges

are modelled by **positive** values of **Radial1**. This gives rise to **barrel distortion** which is sometimes seen in wide angle lenses. The image below shows this distortion **added to** (using **Mode: Add**) an image by various **positive** values of **Radial1**.

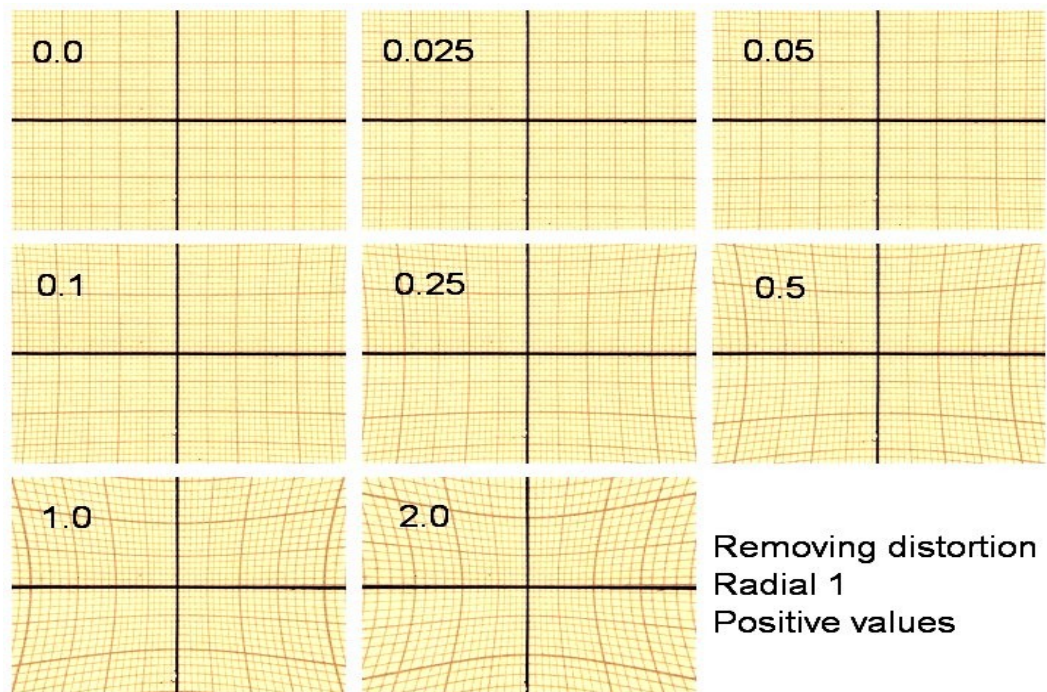


If a lens has a magnification that **increases** as you move away from the center, you get what is called **pincushion distortion**. The image below shows this distortion **added to** an image by various **negative** values of **Radial1** (sometimes seen in higher focal length lenses).

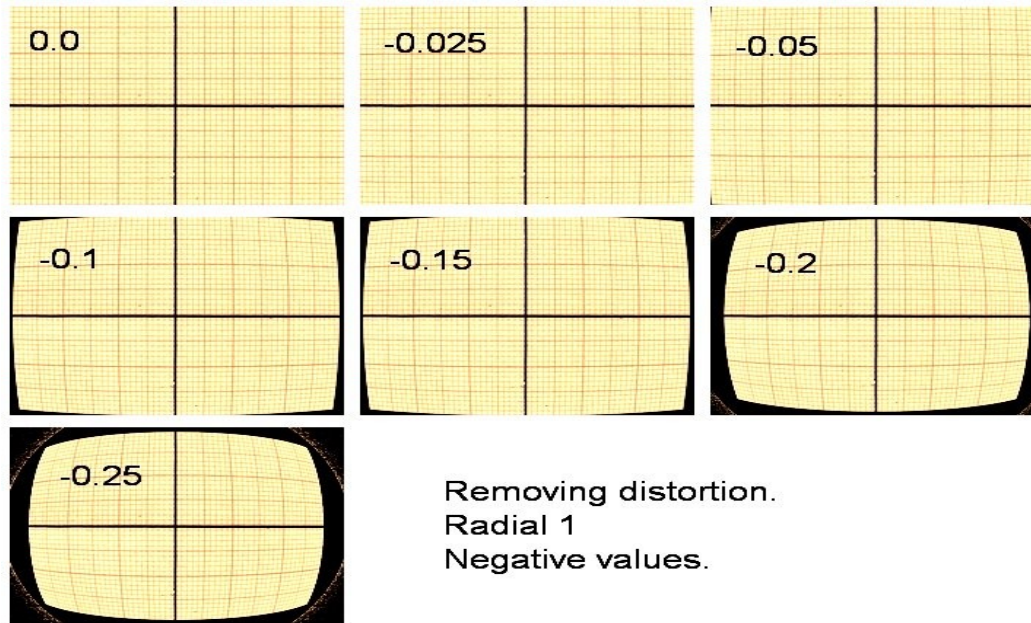




**Removing distortion** using **Mode: Remove** is designed to "undo" the distortion introduced by **Mode: Add**. To the extent that **Mode: Add** accurately models the distortion of real lenses, **Mode: Remove** can remove the distortion introduced by real lenses. **LensCorrect** is designed so that using the same value for **Radial1** (and the other parameters) in two applications of **LensCorrect** -- **Mode: Add**, followed by **Mode: Remove** applied to the result of the **Mode: Add** -- will reproduce the original image. (The same applies to a **Mode: Remove** followed by a **Mode: Add**). There is **one caveat with this**, however: adding or removing distortion may "shrink" the image so that parts of it become black (because the distortion "brings in" areas that are outside the original image). This obviously cannot be undone (although the black regions can be trimmed away using the **Magnify** control). Here is the effect of positive values of **Radial1** in **Mode: Remove**:



The effect of negative values of **Radial1** in **Mode: Remove** is:



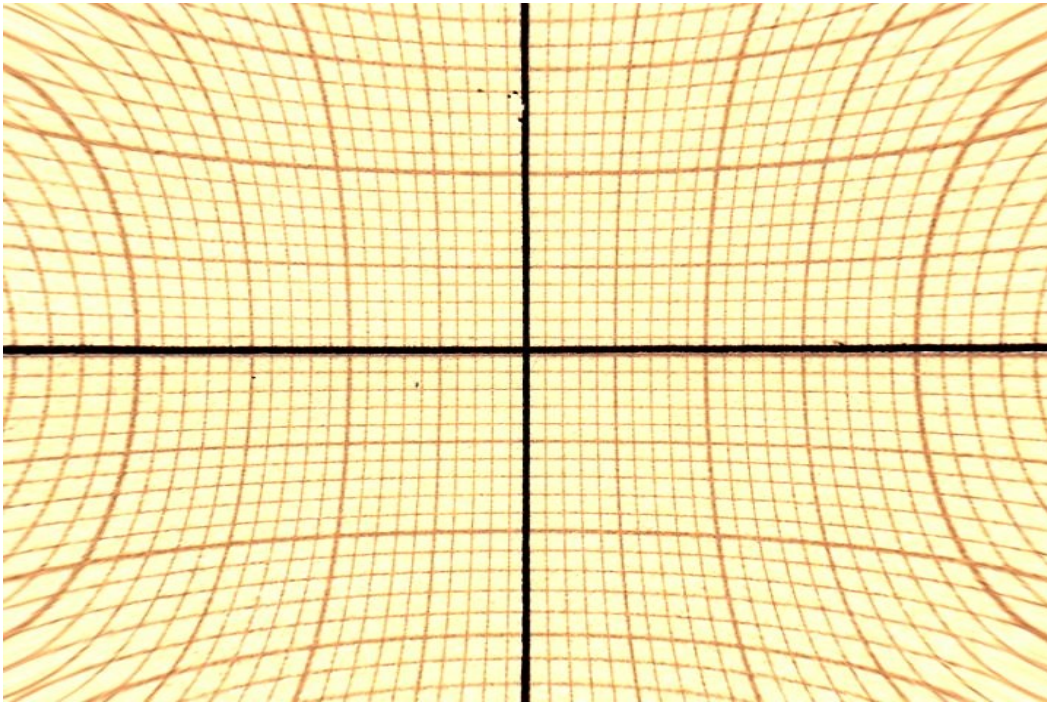
One interesting thing to note in the above images is that for extreme distortions — for example adding with **Radial1** set to  $-1.0$  — single features in the input image appear in more than one place in the distorted image. In this case vertical and horizontal lines are mapped into ellipse-like shapes in the distorted picture. A consequence of this is that such extreme distortions cannot be fully undone, since there is no longer a "unique mapping" back from the distorted image to the original image. The practical consequence of this is that **removing** extreme distortions will give rise to a halo of points near the edges of the output image where the removal algorithm cannot find a unique location on the input image that corresponds to an output pixel. This "things appearing in more than one place" happens with real lenses (or other bits of glass) that cause very great distortion.

**Aside:** In fact, mathematically, the algorithm that removes distortion is much "harder" than the one that adds it. The addition process has what is called a closed-form solution. The removal process requires an iterative, approximate solution.

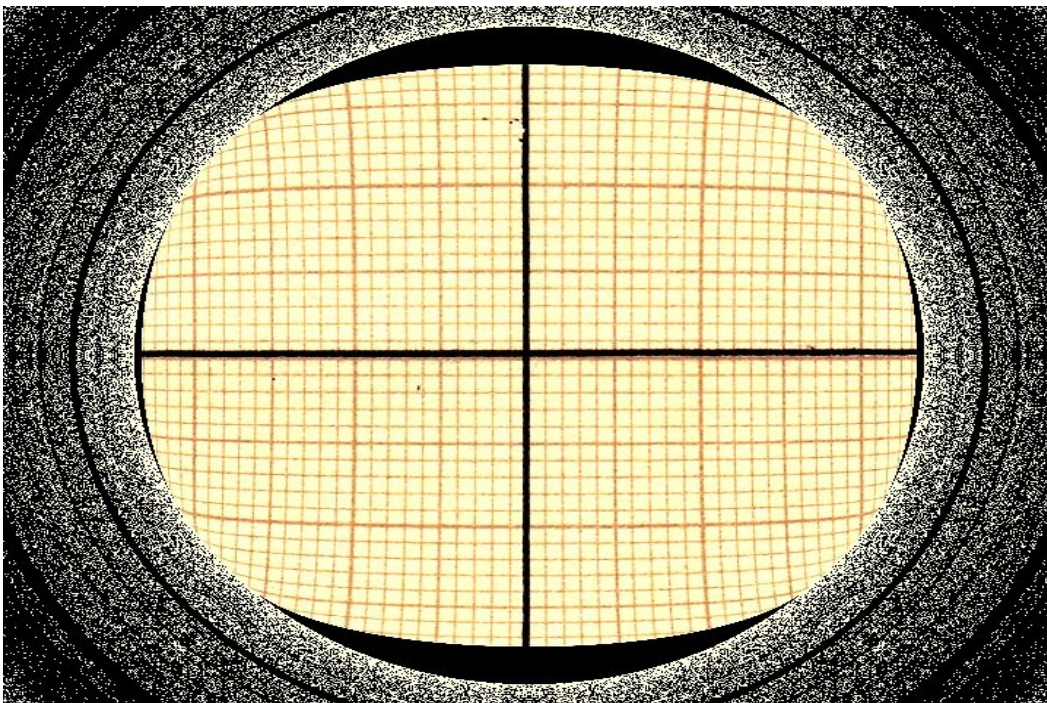
The **Radial2** parameter is a "second order" symmetric distortion that is proportional to the distance from the lens center raised to the power four. You should try to remove as much distortion as possible with **Radial1** before adding any **Radial2** contribution. I.e. try to make straight lines straight with **Radial1**. **Radial2** has the most effect near the edges of the image.

The (usually not very great) distortion of real lenses can usually be corrected for by **Radial1**, possibly with a little **Radial2** added in. Some lenses also have an extra sort of distortion which depends on how far you move along the x-axis only (or the y-axis only). This is usually only significant with anamorphic lenses which are designed to "squeeze" the image along one axis. Below is an image which exhibits both radially symmetric and asymmetric distortion. Not that you can tell this easily just by looking at it!



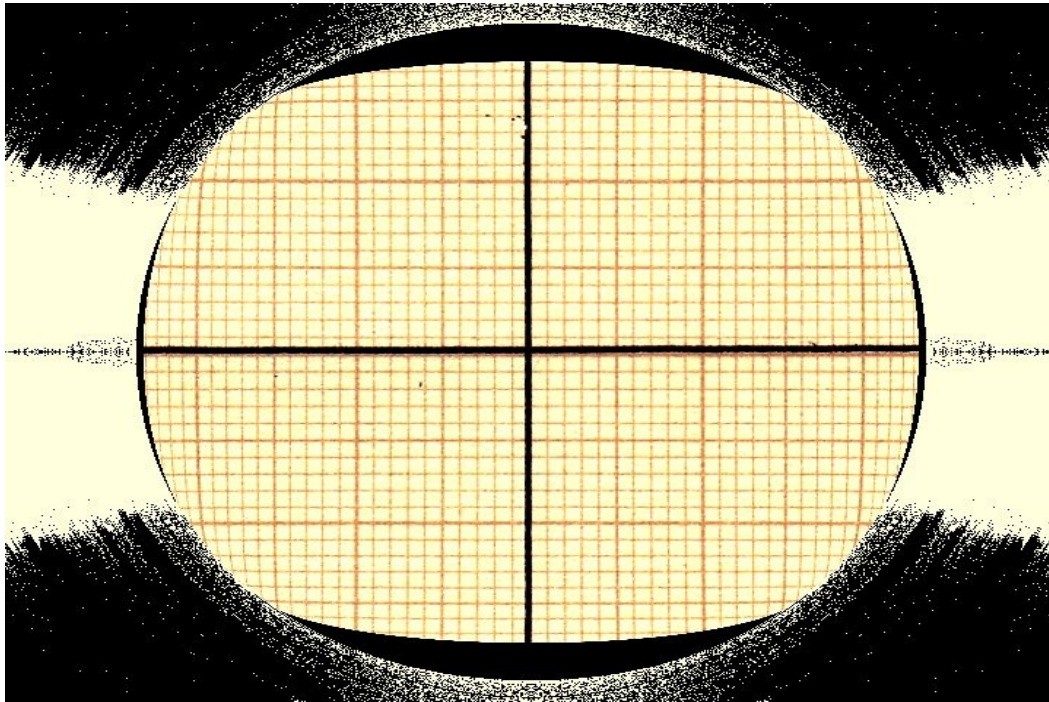


We can use **Radial1** and **Radial2** to remove the radially symmetric components of the distortion. Shown below is the result of doing this:



This was done with **Auto Remove**, in fact. You can see that both the vertical and horizontal lines still show some curvature, however. We can try to correct for this by playing with the **XCurrvature** and **YCurvature** controls which can **independently** adjust curvature in the x and y directions. Here is the result of doing that:

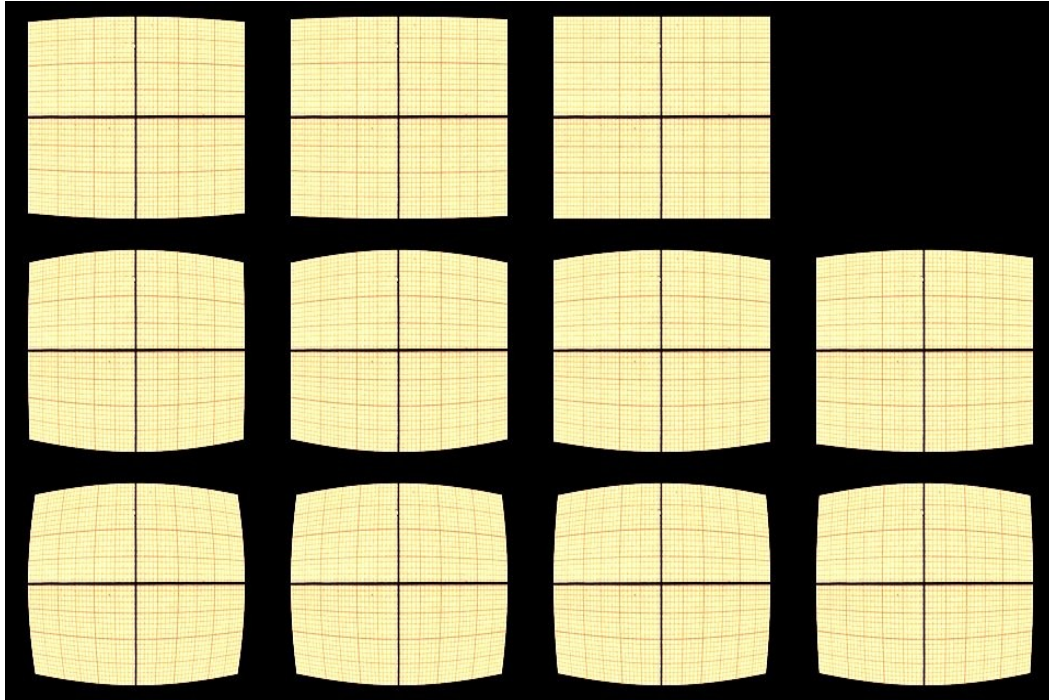




Finally, we can use the **Magnify** control to trim off the parts of the image which had been irretrievably lost by the distortion:



One final note on the asymmetric distortions: these are **in addition** to the radial distortions — you cannot add or remove an asymmetric distortion in the absence of any radial distortion. The image below shows the progressive removal first of curvature in the X direction (**XCurvature**) which straightens the verticals while leaving the horizontals untouched, then the progressive removal of curvature in the Y direction (**YCurvature**) which straightens the horizontals.

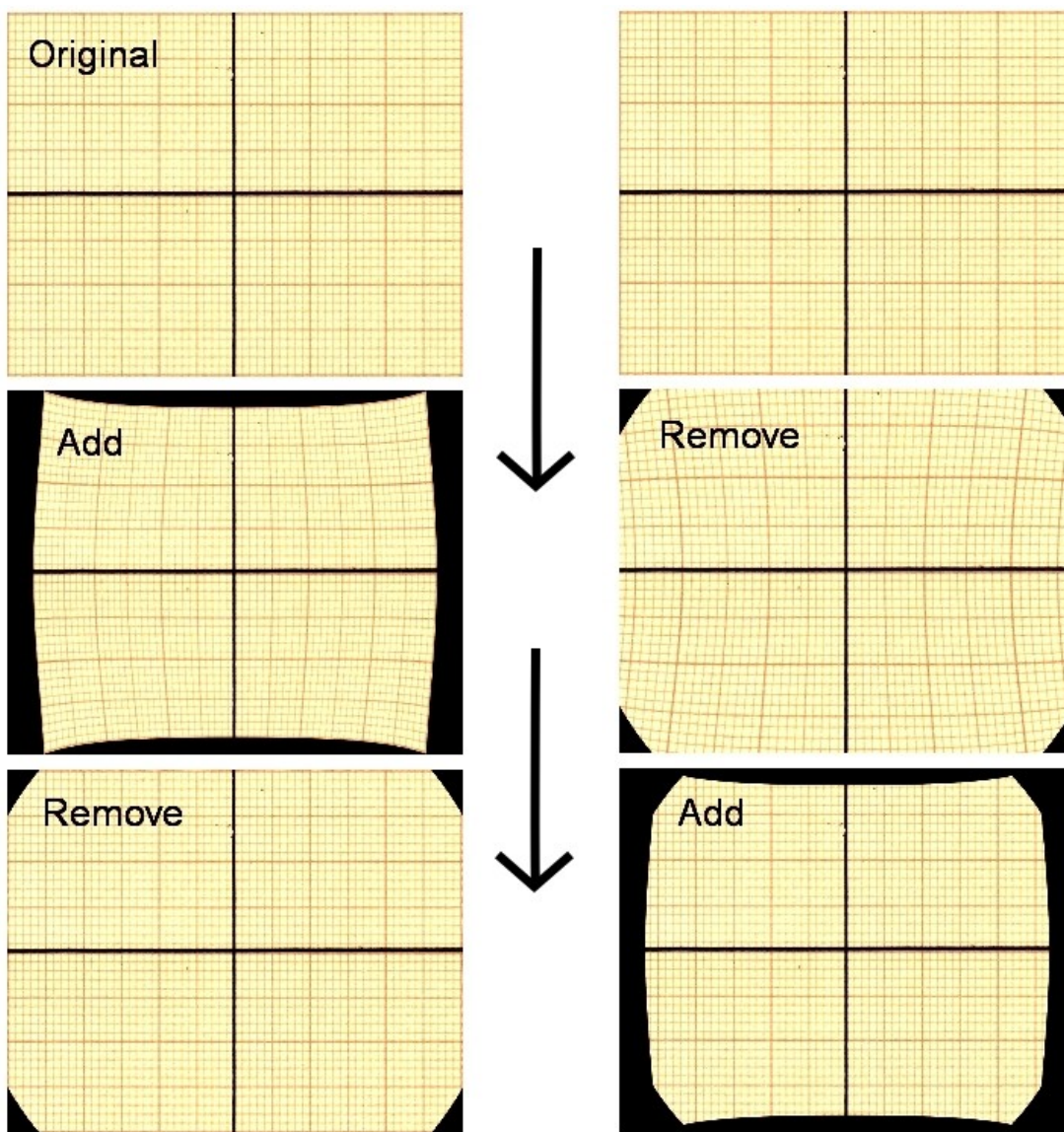


---

**Notes on Removing Distortion and Subsequently Adding Back the Same Distortion**

As discussed above, a **LensCorrect** set to **Mode: Add** will put back the distortion removed by a **LensCorrect** set to **Mode: Remove** if the distortion parameters (**Radial1**, **Radial2**, etc.) are set to the same values in both **LensCorrect** instances. You can also do the **Mode: Remove** then the **Mode: Add**. So long as you use the same values for the distortion parameters in the **Add** as you use in the **Remove** you will end up with the original image. An example of the equivalence of **Add/Remove** and **Remove/Add** is shown below:





Radial1 = 0.41436  
 Radial2 = -0.3139  
 XCurvature = 0.37016  
 YCurvature = -0.7182

If the original image was distorted and the first **LensCorrect** instance is adjusted to *remove* the distortion, then the second **LensCorrect** instance will restore the original distortion. If you feed *undistorted* footage into the second **LensCorrect** instance, it will acquire the distortion exhibited by the original image you fed into the first **LensCorrect** instance. This lets you *match* other footage to the distortion you have in the original image.

There is one major issue to bear in mind, however. **LensCorrect** operates with fixed resolution input and output images — the input and output are the same resolution. Since adding or removing distortion can require access to image data *outside the input image bounds* some of the output



image can be "missing". Usually this will come out as black (although there may be a halo of points that in fact do map back to the input image beyond the black region when removing large distortions).

**LensCorrect** incorporates a naive way of overcoming this problem – the **Magnify** control which can be used to "zoom in" to the result image until the black parts that come from outside the input image are trimmed off. You will often need to do something more sophisticated using other tools in your compositing system to trim away these areas.

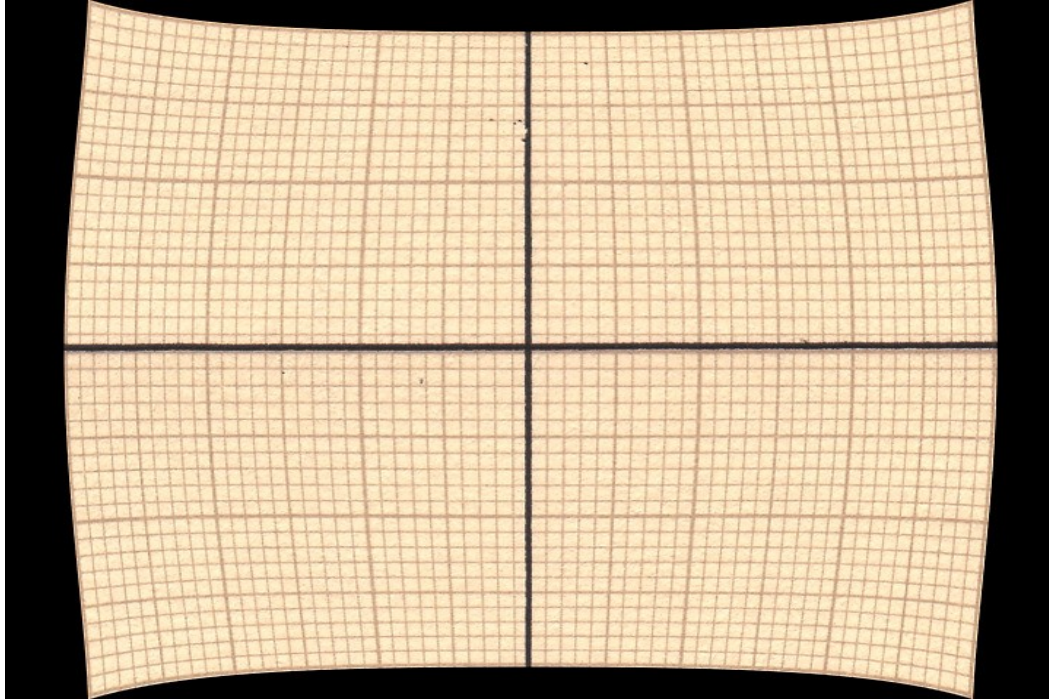
In fact, in real applications, you may have to work with resized (larger) original images at a resolution that ensures that the "off input" regions do not appear in the final output (which you will have to "cut out" of the processed images from **LensCorrect** and whatever else you may be doing).

If you are matching computer generated imagery to real footage, one approach might be to use **LensCorrect** to remove and add back the original distortion, as described above. But you could then use the second **LensCorrect** instance (the one that adds back the original distortion) to distort the CGI footage to match the original. The advantage of distorting the CGI footage is that it is often possible to generate this relatively easily at a size that allows for the "margins" needed by the distortion process.

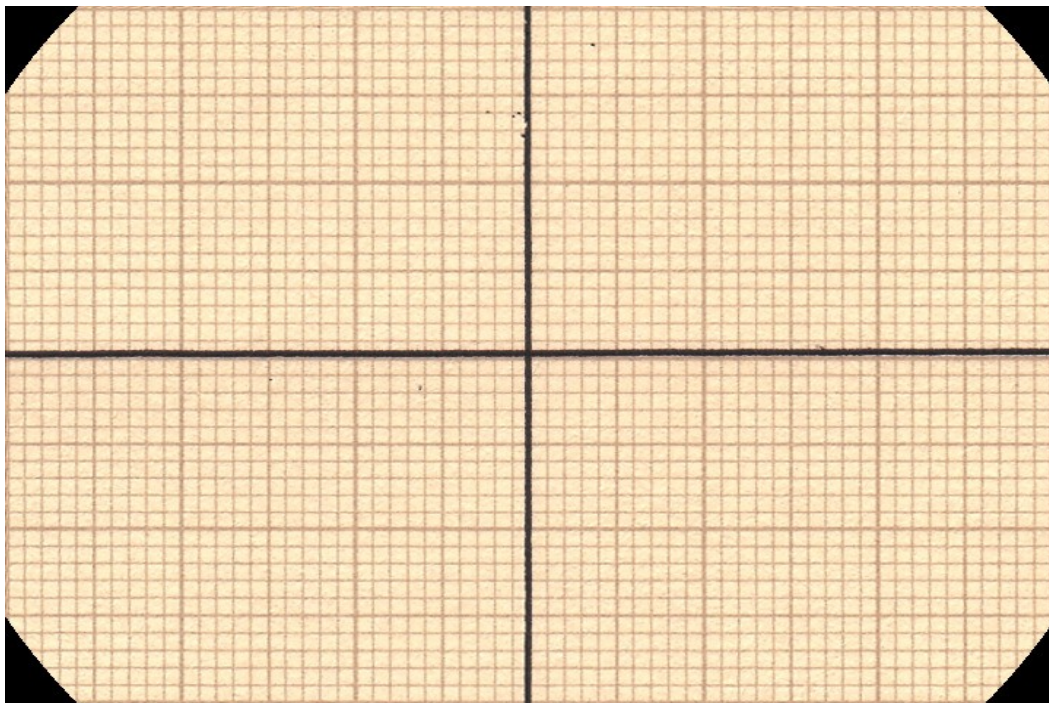
Another matter to bear in mind is that removing then adding back distortion involves two resampling processes applied to the original footage, whereas applying distortion to CGI imagery involves only one resampling process. Since each resampling involves some (hopefully slight!) loss in quality, the fewer resamplings, the better. It is also possible (if expensive) to generate CGI at arbitrarily high resolutions to compensate for any quality loss due to resampling (say at twice the resolution at which the live action footage was scanned, plus margins). Generally, live action footage is acquired at a fixed resolution, and resizing it to create the required "margins" for distortion correction is (yet another) resampling operation that must (slightly) reduce the image quality.

Below is an example of matching undistorted footage to distorted footage. First we see the distorted original footage:

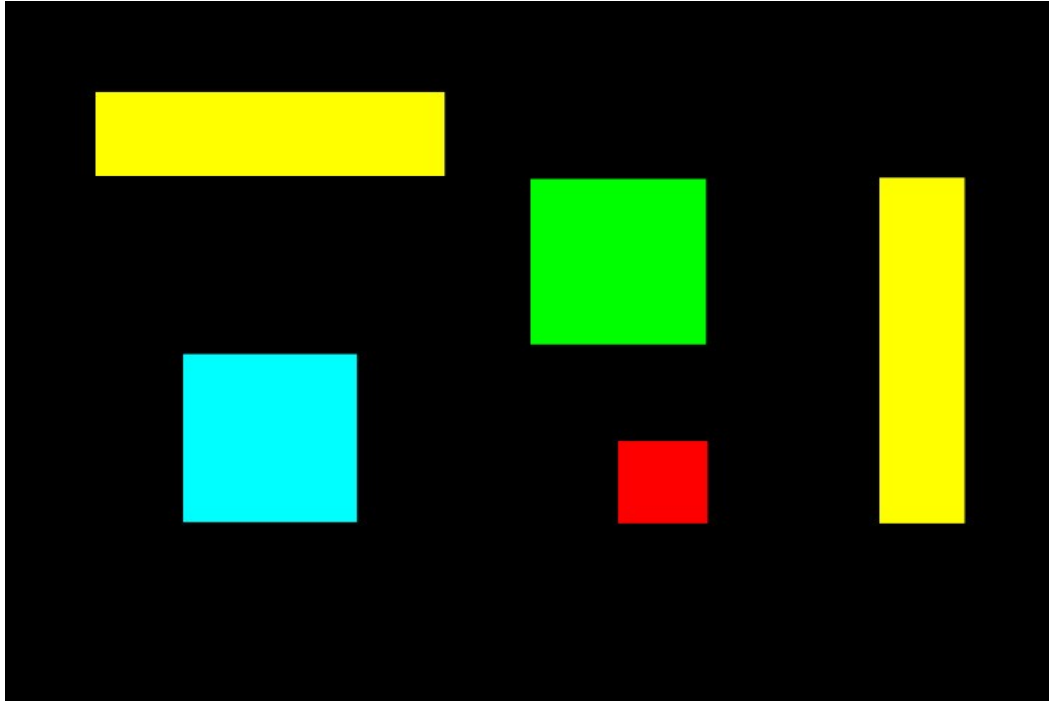
#### 4 SpeedSix.LensCorrect (Discreet Lenz Raptor)



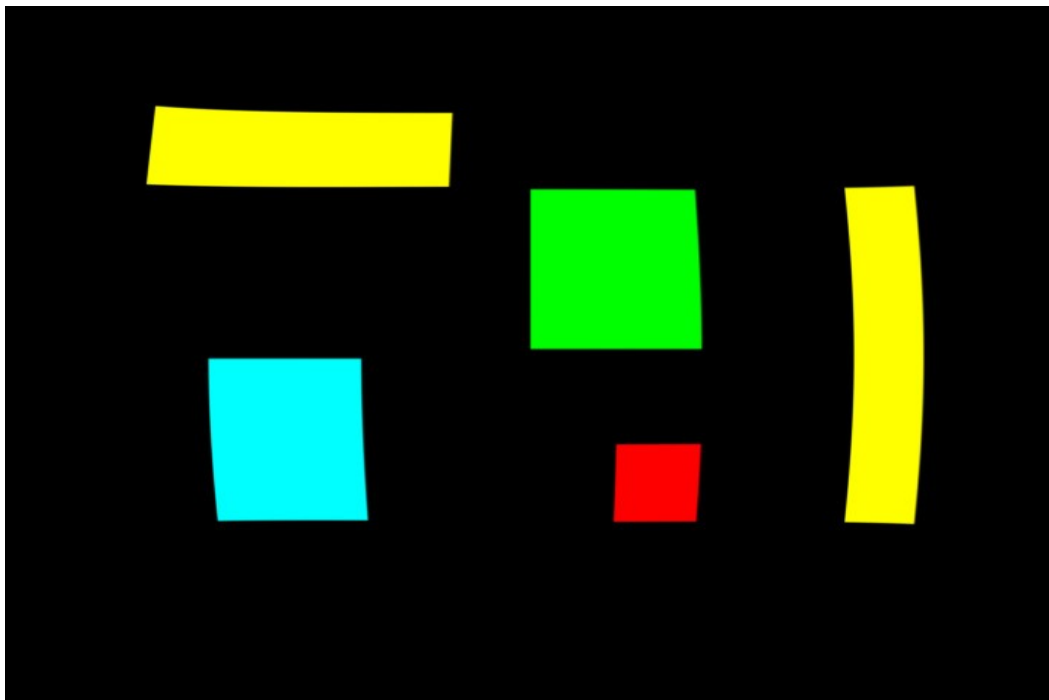
We feed this in to an instance of **LensCorrect** and either manually adjust the parameters or use **Mode Auto Remove** to obtain a satisfactorily undistorted image:



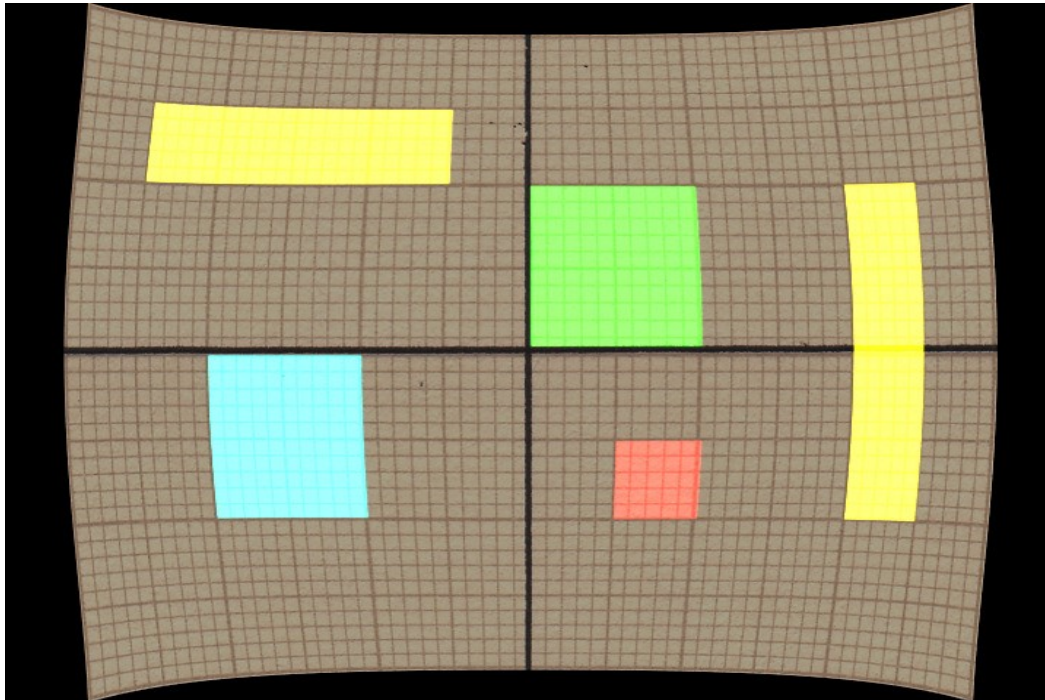
This tells us the parameters we need for another instance of **LensCorrect** set to **Mode: Add** (or **Mode: Remove** if the distortion was removed using **Mode:Auto Remove** — see the note below). This will add back the original distortion. It will add it to any footage that is fed in. Below we see an example of some CGI footage that we want to eventually composite with the original footage. It is undistorted, of course:



After feeding this into the instance of **LensCorrect** set up to add back the distortion found in the original footage, we get this image:



If we composite (or blend here) the distorted CGI and the original distorted footage, we see that they line up correctly:



**One final note:** As mentioned above, **Mode: Auto Remove** — in which **LensCorrect** semi-automatically removes distortion by calculating parameters that straighten line features supplied by the user — works effectively as **Mode: Add**. You need to use the parameters found by **Mode: Auto Remove** in a subsequent **Mode: Remove** instance of **LensCorrect** to add back the original distortion.

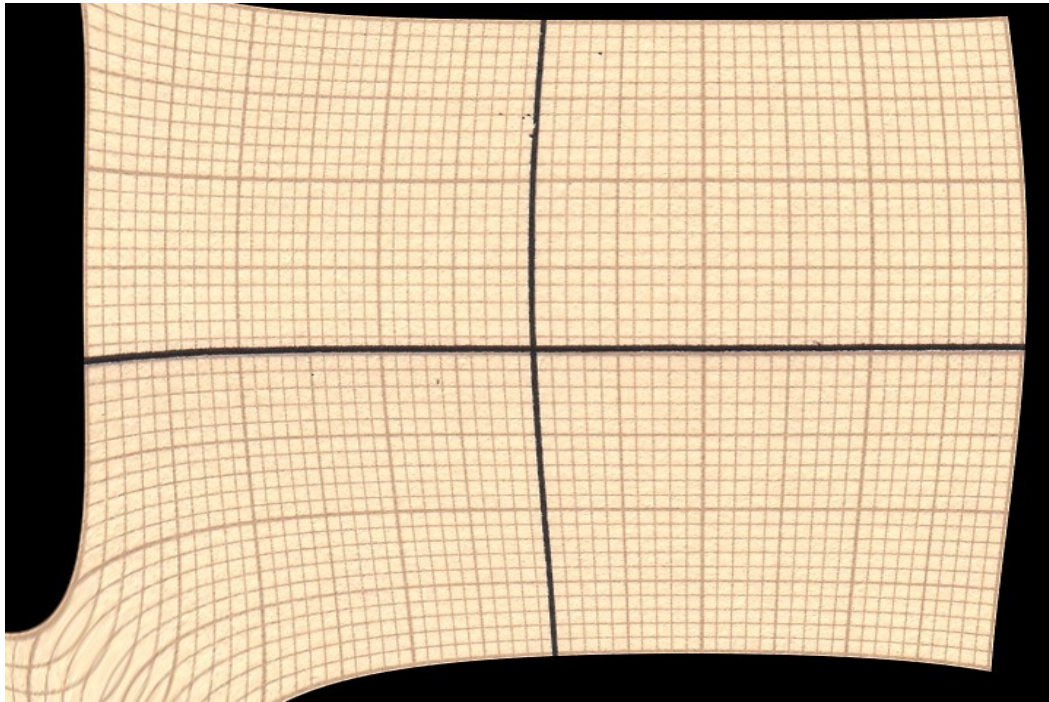
---

### **Lens Centre and Squeeze Factor**

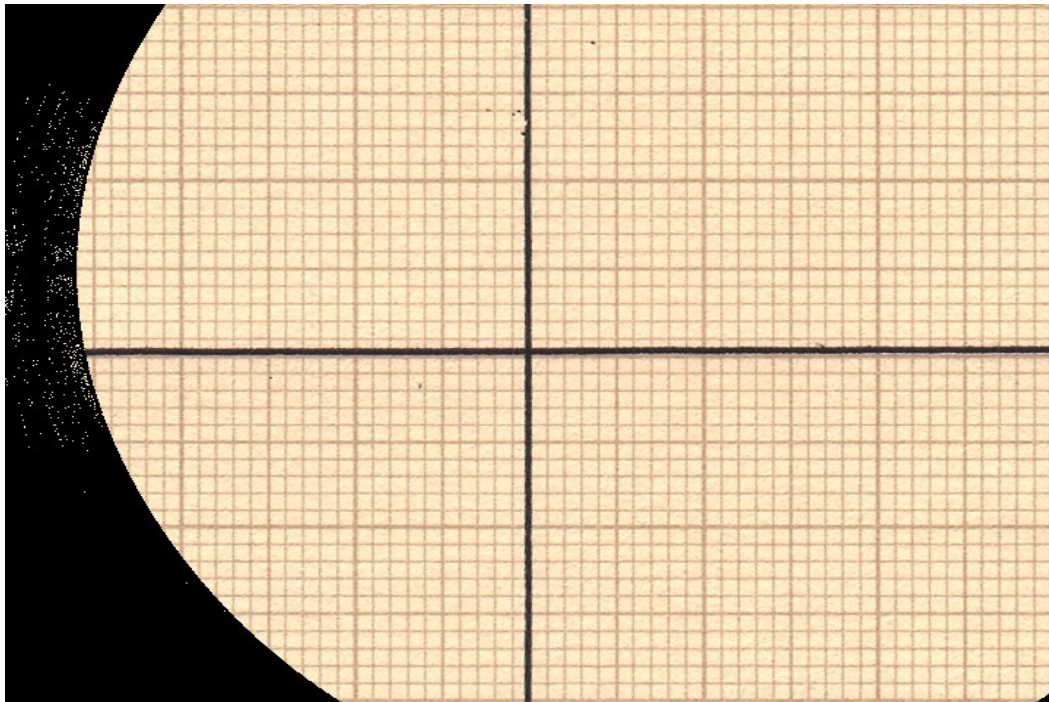
The **Lens Center** sets the position of the *optical centre* of the lens (it is specified in normalized image coordinates (0,0) bottom left of the image and (1,1) top right). For any good lens and camera system, the optical center should be almost exactly in the middle of the image plane (and hence image). So the **Lens Centre** will almost always be (0.5,0.5). You can change it to anywhere on the image, however.

There is one nasty side-effect, though. As you move the lens centre away from the image centre, more and more of the output image is likely to come from "off" the input image — and hence be black. Below is an example of an image distorted with **Lens Centre** set to (0.65,0.38):





And here is the result of un-doing that distortion:



As you can see, we end up with an undistorted picture. But we also end up with a lot of black. This is inevitable.

**Squeeze** should be set to the "horizontal squeeze factor" of the input image (including contributions

#### 4 SpeedSix.LensCorrect (Discreet Lenz Raptor)

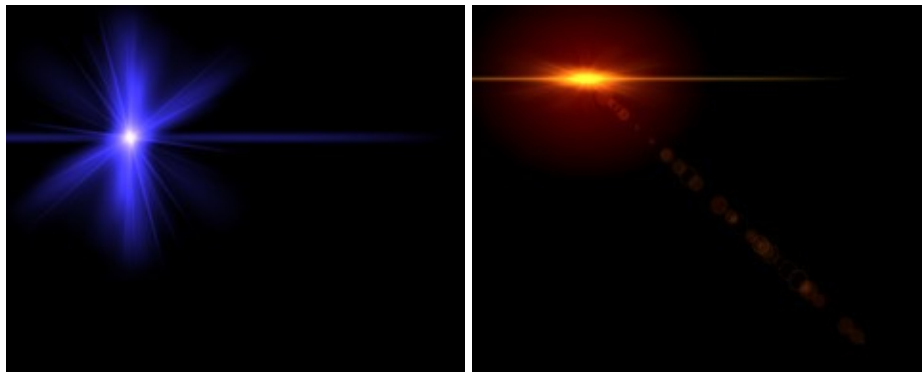
from anamorphic "squeeze" lenses and any "non-square pixel" image formats). If you are not using an anamorphic lens, it is usually satisfactory to leave it at 1.0. Adjusting this control on an otherwise undistorted image will have no effect. It is used internally by **LensCorrect** to compensate for non-uniform scales in the X and Y directions. Since the most important lens distortions depend on distance measured radially from the lens centre, **LensCorrect** needs to know about these axis scale factors so it can calculate radial distances correctly.

## 5 SpeedSix.LensFlare (Discreet Lenz Raptor)

---

### PURPOSE

Lens flares and lens spots before the eyes. This effect simulates some of the effects seen when a very bright light source is in or near the field of view of a camera. These effects are caused by some of the incoming light not being directed to a point on the image but rather being reflected around the internal optical and mechanical components of the camera system. This light finally contributes to the image in the form of "beams" radiating out from the light source, rings and disks caused by geometrical elements of the camera (some circular, some polygonal – e.g. the iris diaphragm) and an unfocussed "glow" which partially washes out the image dependent on distance from the light source location (and quite possibly over the whole of the image). The "un-imaged light" that causes these effects is frequently dispersed in wavelength to create many unexpected colours across the spectrum.



### INPUT CLIPS

**1: Input Clip** : A clip over which lens flares are to appear.

**2: Light** : A clip that can optionally modulate the light source colour. As the light source position moves over this clip, the colour under the light source can be multiplied by the light source colour. If the the **Light Clip Fraction** control is set to 1.0 all the light source colour will come from this clip. Otherwise it will be a blend of the light source colour and the light clip colour.

**3: Obscure** : A clip that can optionally modulate the light source brightness. As the light source position moves over this clip, the fraction of the light source area covered by white in this clip is evaluated and used to control the light brightness. If the whole of the light source area is covered by white areas of this clip, the brightness is zero. If none of the light source area is so covered, the brightness is as set by the light brightness control. Clip values other than white will be partially transparent as evaluated by their luminance.

**4: Track** : A clip that can optionally be used as the input to the (optional) light position tracking function. The light position tracker is a simple tracker which may benefit from having a simple input clip – e.g. a blob indicating where the light source is. If this clip is not used, the Input Clip is used as the light position tracker input. Please see the **Tracking Notes** section below for additional guidance.

## ***LensFlare* CONTROL PAGE**

**Cool** (List Box **Options:** Base | Spikey | SmallBlue | Cute | SoftBlue | StarBlue | BlueRing | PurpleHaze | Halo, **Default:** Base)

Pre-built flare. Sets all controls to get a particular sort of flare look. The presets are all designed using a black background. If you use them over another colour try using the **Light Brightness** and **Contrast** controls on page one to balance the result to suit. See the **Built-in Preset Effects** section to see pictures showing what these look like.

**LightPos** (Position **Default:** 0.27,0.64)

Light source position.

**LightBright** (Number **Min:** 0.00, **Max:** 5.00, **Default:** 0.50)

Brightness of the light source. This may be modified by the area of the light source (determined from its radius set by the LightRad control) that is obscured by solid objects in the **Obscure Clip**.

**LightCol** (Colour Box **Default:** R: 65535, G: 65535, B: 65535, A: 65535)

Colour of the light. This may be changed by a **LightClip** if desired (you will also need to set **LightClipFrac** to something more than 0.0).

**LightClipFrac** (Number **Min:** 0.0, **Max:** 1.0, **Default:** 0.0)

Fraction of the light colour contributed by the light clip.

**Contrast** (Number **Min:** 0.01, **Max:** 10.0, **Default:** 1.0)

Contrast of the flare components

**Warm** (List Box **Options:** Blast | Burst | Smoothy | Dawn | Golden | RedDwarf | Hot | RainbowStar | Pretty, **Default:** Blast)

Pre-built flare. Sets all controls to get a particular sort of flare look. The presets are all designed using a black background. If you use them over another colour try using the **Light Brightness** and **Contrast** controls on page one to balance the result to suit. See the **Built-in Preset Effects** section to see pictures showing what these look like.

**Use Obscure** (Checkbox **Default:** Off)

Use the **Obscure Clip** to control the light brightness. The fraction of the light source area (determined via the light source radius **LightRad**) obscured by the **Obscure Clip** is determined and used to scale the light brightness. The **Obscure Clip** is treated as a matte, with full white being fully obscuring and black being transparent. If **Directional Obscure** is turned on, the **Obscure Clip** is used in a slightly different way. In that case, the light brightness depends on the angle around the light. The brightness is inversely proportional to how much white in the **Obscure Clip** lies along the angle within the light radius. This directional brightness variation affects only the Flare beams. No



other element is affected.

**DirObscure** (Checkbox **Default:** Off)

Use the **Obscure Clip** to control the light brightness as a function of angle around the light. This directional brightness will only affect the flare beam elements and only if **ObscureOn** is selected.

**LightRad** (Number **Min:** 0.001, **Max:** 1.000, **Default:** 0.1)

Radius of light source itself. Used to determine obscuration in conjunction with the **Obscure Clip**. It has no effect if the **Obscure Clip** is not being used.

**Dither** (Number **Min:** 0.00, **Max:** 100.00, **Default:** 0.0)

Add a dither to break up any visible mach banding.

**FBackFill** (Checkbox **Default:** On)

Softens the zone of the flares.

**GBackFill** (Checkbox **Default:** Off)

Softens the zone of the flares.

**More** (List Box **Options:** Halogen | Sharp | Graphic | SmallWhite | SearchLight | CamCorder | 105Prime | Hi8 | SpotLight, **Default:** Halogen)

Pre-built flare. Sets all controls to get a particular sort of flare look. The presets are all designed using a black background. If you use them over another colour try using the **Light Brightness** and **Contrast** controls on page one to balance the result to suit. See the **Built-in Preset Effects** section to see pictures showing what these look like.

**Tracking** (Checkbox **Default:** Off)

Turn on to use the built-in tracking facility.

**Track Light** (Checkbox **Default:** Off)

Use the built-in tracking facility to attempt to track the light source position. The tracker will use the **Input Clip** for its calculations by default. You can choose to use the **Track Clip** instead by turning on the **Use Track Clip** checkbox. The tracking template is always centered on the current light source position when a tracking template "snapshot" is taken. This happens when you press the **Set Track Template** button and also whenever tracking is turned on, or the template radius is changed or the input image resolution changes. Please see the **Tracking Notes** section below for additional guidance.

**UseTrackClip** (Checkbox **Default:** Off)

If selected, use the **Track Clip** as input to the light source position tracker, instead of the **Input Clip**. Please see the **Tracking Notes** section below for additional guidance.

**TrackTemplateR** (Number **Min:** 0.001, **Max:** 0.100, **Default:** 0.01)

Size of the tracking template region. This is the length of the sides of a "square" area. This area is used to take a tracking template "snapshot" from either the Input Clip or the Tracking Clip. The

conditions under which a "snapshot" is taken are explained in the description of the TrackLightPos control. The "snapshot" is always taken at the current light position.

**TrackSearchS** (Number **Min:** 0.001, **Max:** 0.200, **Default:** 0.05)

Size of the tracking search region. This is the length of the sides of a "square" area inside which the tracker will look for the best match with the tracking template "snapshot" (either in the Input Clip or the Tracking Clip according to the selected tracking image source). The position of the best match inside this region is estimated with sub-pixel resolution and used to update the current light source position when tracking is turned on. Please see the **Tracking Notes** section below for additional guidance.

**SetTrackTemplate** (Pushbutton)

Set the light source position tracker template. A "snapshot" is taken of the **Input Clip** or the **Track Clip** (according to the **UseTrackClip** control), centered on the current light position. This "snapshot" is then used to search for any movement of the light source in subsequent frames. Changing the template size, the clip to try to track, turning tracking off and then on again, or changing the resolution of the images being processed will also result in a new template "snapshot" being automatically taken. Please see the **Tracking Notes** section below for additional guidance.

**TurnOffAll** (Checkbox **Default:** Off)

Turn off any active flares, glows, rings and secondaries to speed up the tracking. Once the track has been set up turn off to re-activate your flares etc. All the controls relating to the effects will disappear so you **MUST** turn this control off again to get them back!

## **Flare CONTROL PAGE**

**Flare** (Checkbox **Default:** On)

Turn on the primary flare element. The primary flare element has many beams radiating from the light source position. These beams fade off with distance from the light source. They have a colour (usually white) which is multiplied by the light source colour. That colour can optionally be blended with a rainbow (or spectrum). The rainbow colours pass through red to violet with distance from the light source. The initial colour and the number of times we pass through the rainbow over the maximum beam length can be controlled.

**Beams** (Number **Min:** 1, **Max:** 1000, **Default:** 30)

Number of beams radiating from the light source center. The flare is made up of one or more such beams. These beams are created at random angles around a circle. This angular distribution is set by **FlareSeed** random number seed control.

If the **FlareSeed** is set to the special value of **0**, then the distribution of beams is **not** random. In that case, the beams are uniformly distributed around the circle. E.g. with six beams would each would have 60 degrees between them.

**FlareColour** (Colour Box **Default:** **R:** 16176, **G:** 14128, **B:** 65535, **A:** 65535)

Base colour of the beams. Spectral colour may be added on top of this – i.e. a colour selected from a rainbow, the colour selected being dependent on distance from the light source position. The amount of spectral colour added in is controlled by the **SpectralFrac** control. The final colour of a

beam at any point is the combination of the base and spectral colour multiplied by the light source colour.

**Bright** (Number Min: 0.00, Max: 5.00, Default: 0.4)

Varies the relative brightness of the flare element. The beam brightness is the light source brightness multiplied by this flare brightness.

**Rotation** (Number Min: 0.0, Max: 360.0, Default: 0.0)

Rotation of the beams about the light source center.

**FlareDistort** (Number Min: 1.00, Max: 5.00, Default: 1.00)

Radial distortion for the flare element. This can be used to make the flare beams lie within a vertically squashed circle (an ellipse!). At 1.0, the flare beams will be un-distorted (lie within a circle).

**Scale** (Number Min: 0.01, Max: 5.00, Default: 1.00)

Overall size scaling factor for the flare element. All the beams are scaled up and down by this factor (about the light source position).

**MinRadius** (Number Min: 0.002, Max: 5.000, Default: 0.41)

Minimum beam radius. Beams will be created that are at least this long (measured from the light source center). If **FlareSeed** is 0, the beams will all be this long.

**MinWidth** (Number Min: 0.002, Max: 0.500, Default: 0.010)

Minimum beam width. This is the narrowest beam that will be created. If **FlareSeed** is 0, the beams will all be this wide.

**StartWidth** (Number Min: 0.10, Max: 5.00, Default: 0.4)

Beam starting point width scale. The beams width at the light source position is the width the beam was created at, multiplied by this scale factor. The width of the beam can change over its length from this width to the end width – the product of the beams creation width and the end point width scale.

**SpecialBright** (Number Min: 1.0, Max: 20.0, Default: 2.0)

Any *special* dominant beams in the flare (you can choose no dominant beam, vertical dominant, horizontal dominant or both vertical and horizontal) will be this much brighter than any other beam in the flare.

**SpecialLength** (Number Min: 1.0, Max: 10.0, Default: 2.0)

Any *special* dominant beams in the flare (you can choose no dominant beam, vertical dominant, horizontal dominant or both vertical and horizontal) will be this much longer than any other beam in the flare.

**SpecialWidth** (Number Min: 1.0, Max: 10.0, Default: 2.0)

Any *special* dominant beams in the flare (you can choose no dominant beam, vertical dominant, horizontal dominant or both vertical and horizontal) will be this much wider than any other beam in the flare.

**FlareHarmonics** (Number **Min:** 0, **Max:** 3, **Default:** 0)

If the Flare beam distribution is **not random**, insert this many extra sets of beams between the **Nbeams** main beams. The first extra set will be half the length of the main beams; the second one quarter the length and the third one-eighth.

**MaxRadius** (Number **Min:** 0.002, **Max:** 5.000, **Default:** 0.94)

Maximum beam radius. This is the longest beam length that will be created.

**MaxWidth** (Number **Min:** 0.002, **Max:** 0.500, **Default:** 0.050)

Maximum beam width. This is the widest beam that will be created.

**EndWidth** (Number **Min:** 0.10, **Max:** 5.00, **Default:** 0.50)

Beam end point width scale. The beams width at its end is the width the beam was created at multiplied by this scale factor. The width of the beam can change over its length from the starting width to this end width.

**SpectralFrac** (Number **Min:** 0.00, **Max:** 1.00, **Default:** 0.00)

Contribution of a colour from a rainbow (or spectrum) to the flare colour. This colour changes with distance from the light source center at a rate controlled by **SpectralCycles**. When added to the flare base colour, it determines the flare colour, which is then multiplied by the light source colour to form the final colour. At 0.0 there is no contribution from a rainbow and the flare colour is the base flare colour.

**SpectralCycles** (Number **Min:** 0.10, **Max:** 10.00, **Default:** 1.00)

Number of cycles of a spectrum that will take place over the **MaxRadius** length from the light source center. The higher this number, the more quickly the rainbow colour will change with distance out from the light source.

**SpectralPhase** (Number **Min:** 0.00, **Max:** 1.00, **Default:** 0.00)

Starting point (i.e. colour) for the spectrum at the light source position.

**FlareSeed** (Number **Min:** 0, **Max:** 7483648, **Default:** 12345)

Random number seed that determines the angular distribution of the beams. Changing this number will create a new set of beams with a different angular distribution. The value **0** is special – it indicates that there is to be **no randomness** in the beam distribution. If there are 6 beams, they will be positioned equally spaced around a circle at intervals of 60 degrees.

**BeamDom** (List Box **Options:** None | Horizontal | Vertical | Both, **Default:** Horizontal)

Dominant beam direction. This can guarantee that a horizontal or vertical beam (or both) is created regardless of the rest of the angular beam distribution. This beam will be as long and wide as is allowed and will be drawn at maximum brightness.

**SymmetricBeams** (Checkbox **Default:** On)

Make all beams symmetrical about the light source center. If **off**, beams start from the light source position. If **on**, beams have their middle at the light source position.

---

## ***Glow Ring* CONTROL PAGE**

**GlowBall** (Checkbox **Default:** Off)

Turn on the primary "glow ball" element. This is a ball shaped glow centered on the light source. You can control its radius and how quickly it fades away. The colour (usually white) can be set, and this is multiplied by the light source colour to find the "glow ball" colour. This colour can be blended with a rainbow (or spectrum) the colour of which varies with distance from the light source, as described for the primary flare element.

**GlowRadius** (Number **Min:** 0.000, **Max:** 5.000, **Default:** 0.150)

Radius of the glow ball (center is the light source position).

**GlowFadeOff** (Number **Min:** 0.000, **Max:** 5.000, **Default:** 0.200)

Length (from the light source center) over which the glow ball fades to zero intensity.

**GlowBright** (Number **Min:** 0.00, **Max:** 5.00, **Default:** 1.0)

Varies the relative brightness of the glow element. The glow brightness is the light source brightness multiplied by this glow brightness.

**GlowColour** (Colour Box **Default:** R: 16176, G: 14128, B: 65535, A: 65535)

Colour of the glow. This colour is multiplied by the light source colour to find the final glow colour.

**GlowProfile** (List Box **Options:** Linear | Hermite | Cosine | Tight | Halo, **Default:** Cosine)

Mathematical function used to control how the glow fades across its **FadeOff** width.

The glow intensity falls off to zero over its **FadeOff** width with a "shape" controlled by this setting.

**GlowDistort** (Number **Min:** 1.00, **Max:** 5.00, **Default:** 1.00)

Radial distortion for the glowball element. This can be used to make the glow lie within a vertically squashed circle (an ellipse!). At 1.0, the flare beams will be un-distorted (lie within a circle).

**Ring** (Checkbox **Default:** Off)

This element is a single ring surrounding the light source position. You can control its radius and how thick it is. You can also control whether it starts abruptly and dies away over its width, builds up over its width and stops abruptly, or is symmetrical in the way it builds up and fades away. You can give the ring its own colour (often white), which is multiplied by the light source colour and optionally blended with a rainbow as described above for the "beam" and "glow ball" elements. The colour selected from the rainbow at any point depends on position within the width of the ring.

**RingRadius** (Number **Min:** 0.000, **Max:** 5.000, **Default:** 0.500)

Radius of the ring (center is the light source position).

**RingFadeOff** (Number **Min:** 0.000, **Max:** 5.000, **Default:** 0.80)

Length (measured from the ring radius, inside, outside or on both sides of the ring radius, depending on ring type) over which the ring fades to zero intensity.

**RingBright** (Number **Min:** 0.00, **Max:** 5.00, **Default:** 1.0)

Varies the relative brightness of the ring element. The ring brightness is the light source brightness multiplied by this ring brightness.

**RingColour** (Colour Box **Default:** **R:** 65535, **G:** 65535, **B:** 65535, **A:** 65535)

Colour of the ring. This colour is multiplied by the light source colour to find the final ring colour.

**RingDistort** (Number **Min:** 0.10, **Max:** 5.00, **Default:** 2.00)

Radial distortion for the ring element. This can be used to make the ring a prolate or oblate ellipse for values less than 1.0 and greater than 1.0. At 1.0, the ring will be circular. This can also be used with polygonal settings – in which case the results are "interesting"!

**Rotation** (Number **Min:** 0.0, **Max:** 360.0, **Default:** 0.0)

Rotation of the ring about its center.

**RingType** (List Box **Options:** OutRing | InRing | SymRing, **Default:** SymRing)

Shape of the ring element. An **OutRing** starts abruptly **RingRadius** distance from the light source center and its intensity falls to zero after **RingFadeOff** distance beyond **RingRadius**. An **InRing** starts **RingFadeOff** distance inside the **RingRadius** and ends abruptly at **RingRadius**. A **SymRing** builds up to maximum intensity at **RingRadius**, beginning from zero at **RingFadeOff** distance inside the **RingRadius** and returning to zero at **RingFadeOff** distance outside **RingRadius**.

**RingSides** (Number **Min:** 0, **Max:** 50, **Default:** 0)

Number of straight sides for the ring element. Any number less than 3 will give a circle. Three will give a triangle, 4 a square, and so on. Commonly seen polygonal shapes in real camera lens flares are pentagons and hexagons (although I've sometimes seen triangles on camcorders).

**RingSpectralFrac** (Number **Min:** 0.00, **Max:** 1.00, **Default:** 1.00)

Contribution of a colour from a rainbow (or spectrum) to the ring colour. This colour changes with distance from the light source center at a rate controlled by **SpectralCycles**. When added to the ring base colour, it determines the ring colour, which is then multiplied by the light source colour to form the final colour. At 0.0 there is no contribution from a rainbow and the ring colour is the base ring colour.

**RingSpectralCycles** (Number **Min:** 0.10, **Max:** 10.00, **Default:** 1.00)

Number of cycles of a spectrum that will take place over the **FadeOff**. The higher this number, the more quickly the rainbow colour will change with distance out across the **FadeOff** region.

**RingSpectralPhase** (Number **Min:** 0.00, **Max:** 1.00, **Default:** 0.88)

Colour the spectrum will have at the **RingRadius**.

**RingProfile** (List Box **Options:** Linear | Hermite | Cosine, **Default:** Hermite)

Mathematical function used to control how the ring fades across its **FadeOff** width. The ring has maximum intensity at its **RingRadius** and falls off to zero over its **FadeOff** width with a "shape" controlled by this setting.

**RingPos** (Number **Min:** 0.00, **Max:** 1.00, **Default:** 0.00)

Set the position of the Ring element along the Secondary elements line. Typically, the Ring will be wanted centered on the light source, but this lets you move it if desired.

**FringeSeed** (Number **Min:** 1, **Max:** 7483648, **Default:** 12345)

Random number seed that determines the appearance of any ring fringeing. Changing this number will create a new set of fringes with a different angular distribution.

**FringeFreq** (Number **Min:** 1.00, **Max:** 10.00, **Default:** 1.00)

Controls the maximum detail in the ring fringeing effect.

**FringeAmp** (Number **Min:** 0.00, **Max:** 1.00, **Default:** 1.00)

Controls how deeply the ring intensity is modulated by the fringeing effect.

---

## **Secondaries CONTROL PAGE**

**Secondaries** (Checkbox **Default:** Off)

Secondary elements are "spots" or "rings" strung out along a "secondary element line". This line starts at the light source position and passes through the center of the image. The number of elements can be controlled. The position of the elements along the line is random and is determined by setting a random number seed specific to secondary elements. The colour of the elements is a controllable blend of the light colour, a base colour for all secondary elements, and a random colour variation, the hue range of which can be controlled.

**Spots** (Number **Min:** 0, **Max:** 300, **Default:** 30)

Number of secondary elements (spots and rings). This number of elements will be created with their parameters randomly varied between limits set by the secondary element controls (on this page). The distribution of parameters is controlled by the **SpotSeed** random number seed.

**SpotSides** (Number **Min:** 0, **Max:** 50, **Default:** 0)

Number of straight sides for the spots and rings. All the secondary elements will have the same number of sides.

**Bright** (Number **Min:** 0.00, **Max:** 5.00, **Default:** 1.00)

Varies the relative brightness of the spot elements. The spot brightness is the light source brightness multiplied by this spot brightness.

**Scale** (Number **Min:** 0.01, **Max:** 5.00, **Default:** 1.00)

Overall size scaling factor for the spot elements. All the spots are scaled up and down by this

factor (about their center positions).

**Rotation** (Number **Min:** 0.0, **Max:** 360.0, **Default:** 0.0)

Rotation of the spots about their centers.

**SpotDistort** (Number **Min:** 0.10, **Max:** 5.00, **Default:** 1.00)

Radial distortion for the spot elements. This can be used to make the spots prolate or oblate ellipses for values less than 1.0 and greater than 1.0. At 1.0, the spots will be circular. This can also be used with polygonal settings – in which case the results are "interesting"!

**SpotMinRadius** (Number **Min:** 0.001, **Max:** 5.000, **Default:** 0.0020)

Minimum spot radius. Spots will be created that have at least this radius.

**SpotMinFade** (Number **Min:** 0.0001, **Max:** 5.000, **Default:** 0.005)

Minimum spot fadeoff distance. Spots will be created that have at least this fadeoff distance.

**SpotHueVar** (Number **Min:** 0.00, **Max:** 1.00, **Default:** 0.50)

This controls how much variation there will be in spot colour about the chosen spot base colour. The colour of a spot is the spot base colour, randomly shifted along the spectrum by up to this much of the full spectrum length, multiplied by the light source colour.

**SpotColour** (Colour Box **Default:** **R:** 16176, **G:** 14128, **B:** 65535, **A:** 65535)

Colour of the spots. This colour is first perturbed to an extent controlled by the **Spot Hue Variance**, then multiplied by the light source colour to find the final spot colour.

**LineLength** (Number **Min:** 0.000, **Max:** 10.000, **Default:** 2.000)

Length of the line along which "secondary elements" will appear. Secondary elements are "spots" or "rings" strung out along this line. The number of elements can be controlled. The position of the elements along the line is random and is determined by setting a random number seed specific to secondary elements. The colour of the elements is a controllable blend of the light colour, a base colour for all secondary elements, and a random colour variation.

**SpotSeed** (Number **Min:** 1, **Max:** 7483648, **Default:** 12345)

Random number seed that determines the distribution of the spots along the secondary element line (which starts at the light source and passes through the center of the image. Changing this number will create a new set of spots with a different distribution along that line.

**SpotsBothSides** (Checkbox **Default:** Off)

Make spots appear along the continuation of the secondary element line through the light source on the side away from the image center. This seems to be seen only rarely in real camera flares – but it does happen sometimes. Please note that turning this on will change the positions of spots (relative to when it was off).

**SpotMaxRadius** (Number **Min:** 0.001, **Max:** 5.000, **Default:** 0.050)

Maximum spot radius. This is the biggest radius spot that will be created.

**SpotMaxFade** (Number **Min:** 0.0001, **Max:** 5.000, **Default:** 0.010)



Maximum spot fadeoff distance. This is the biggest fadeoff spot that will be created.

**SpotBrightVar** (Number **Min:** 0.00, **Max:** 1.00, **Default:** 0.50)

This controls how much variation there will be in spot brightness from full brightness. If this is 0.0, all spots will be maximally bright. If this is 1.0, spot brightness will vary from maximally bright to zero brightness (invisible).

**SpotDiskProb** (Number **Min:** 0.00, **Max:** 1.00, **Default:** 1.00)

This controls the relative likelihood of creating a **Disk** (filled circular or polygonal) secondary element, as opposed to a ring of some kind. The probabilities specified for each type of disk or ring are summed, and the chance of choosing a **Disk** when the secondary elements are created is the **SpotDiskProb** divided by that sum. The actual size of this number therefore isn't important. What matters how big or small it is compared to the other element probability specifications (**SpotOutRingProb**, **SpotInRingProb** and **SpotSymRingProb**).

**SpotOutRingProb** (Number **Min:** 0.00, **Max:** 1.00, **Default:** 0.00)

This controls the relative likelihood of creating an **OutRing** secondary element, as opposed to a disk or some other kind of ring. The probabilities specified for each type of disk or ring are summed, and the chance of choosing an **OutRing** when the secondary elements are created is the **SpotOutRingProb** divided by that sum. The actual size of this number therefore isn't important. What matters how big or small it is compared to the other element probability specifications (**SpotDiskProb**, **SpotInRingProb** and **SpotSymRingProb**).

**SpotInRingProb** (Number **Min:** 0.00, **Max:** 1.00, **Default:** 0.00)

This controls the relative likelihood of creating an **InRing** secondary element, as opposed to a disk or some other kind of ring. The probabilities specified for each type of disk or ring are summed, and the chance of choosing an **InRing** when the secondary elements are created is the **SpotInRingProb** divided by that sum. The actual size of this number therefore isn't important. What matters how big or small it is compared to the other element probability specifications (**SpotDiskProb**, **SpotOutRingProb** and **SpotSymRingProb**).

**SpotSymRingProb** (Number **Min:** 0.00, **Max:** 1.00, **Default:** 1.00)

This controls the relative likelihood of creating a **SymRing** secondary element, as opposed to a disk or some other kind of ring. The probabilities specified for each type of disk or ring are summed, and the chance of choosing an **SymRing** when the secondary elements are created is the **SpotSymRingProb** divided by that sum. The actual size of this number therefore isn't important. What matters how big or small it is compared to the other element probability specifications (**SpotDiskProb**, **SpotOutRingProb** and **SpotInRingProb**).

---

## **Tracking Notes**

Tracking can be used to have the light source position (**LightPos**) follow some feature in your **input clip**. Sometimes, you may want to follow something in another clip (**track clip**) for reasons discussed below.

Setting up to track something in your **input clip** or in an auxiliary **tracking clip** is easy if a few simple principles are followed.

### When should you use a tracking clip?

The tracker built in to LensFlare is fairly simple (although it is sub-pixel accurate) and if there isn't a feature at the desired light source position in the input clip that looks easy to track, you may want to generate an auxiliary tracking clip from the input clip that may be more "trackable".

For reliable tracking, you generally want something with decent contrast and something that looks more or less like a corner or a cross. However, repetitive patterns (e.g. chicken wire fences) might give problems. The tracker works in monochrome, so contrast needs to be assessed with that in mind.

Apart from increasing contrast, another application for an auxiliary tracking clip is to add an offset. The built-in tracker does not have an "offset tracking" capability. If there is a suitable object for tracking that follows the desired light source position, but is not *at* that position, you could take the input clip and shift it with some host platform function so that the trackable feature is positioned where you want the light source, using the shifted input clip as the track clip.

### Recommended steps to set up and use the tracker

Here is a recommended procedure.

- Turn off **Track Light** if it is on.
- Turn off all the lens flare effects – **Flare, GlowBall, Ring, Secondaries**.
- Position the **LightPos** so the center as shown on the overlay is over the feature you want to track. As explained above, this must also be the light source position, as the tracker has no offset facility. One way of simulating an offset facility is to shift the input clip so a trackable feature lies at the light source position, and use that shifted clip as the **Track Clip**. If you do this, turn on the **UseTrackClip** control.
- Turn on **Track Light**. This records what is under the light location as a tracking template which the tracker will subsequently attempt to find every time a frame is processed.
- Adjust the **TrackTemplateRadius** using the purple box overlay to enclose the feature you intend to track. This will update the tracking template as you make adjustments.
- Adjust the **TrackSearchSide** using the green box overlay to cover the region in which the feature being tracked will move from one frame to the next. This needs to be big enough to accommodate the motion of the feature, but ideally not much bigger – the bigger it is, the slower the tracker will run.
- Allow **animation** or **key framing** on the **LightPos** control using the host system method of doing this.
- Play through the clip. Hopefully, the tracker will follow the feature of interest, creating an animation path for the **LightPos** control.
- Turn off **Track Light**.
- Return to the first frame of the clip.
- Turn on your desired lens flare effects.
- Play through or render the clip. The **LightPos** will follow the animation path generated during tracking.

If the tracker loses track at some frame, stop the play through. Then:

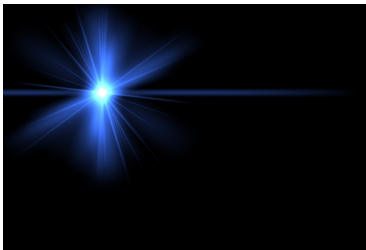
- Turn off **Track Light**.
  - Move to the frame where the track was lost.
  - Move the light position manually to the correct location.
  - Turn on **Track Light**
  - Modify **TrackTemplateRadius** and **TrackSearchSide** to values you think appropriate.
  - Start to play through again.
- 

### **Built-in Preset Effects**

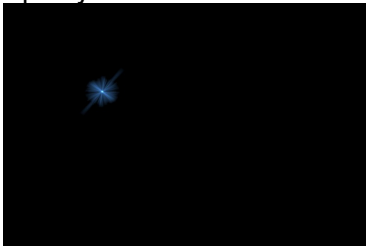
To help you get started, many built in presets are supplied in **LensFlare**. You can choose one of these from any of the three list controls: **Cool**, **Warm** and **More Presets**. You can then adjust the controls from the starting point given by the preset to get the result you want. Here are pictures of each of the presets:

#### Cool

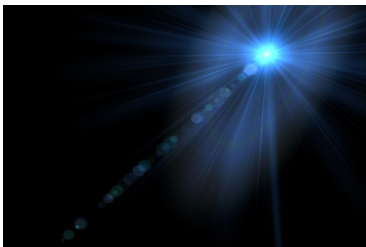
- Base



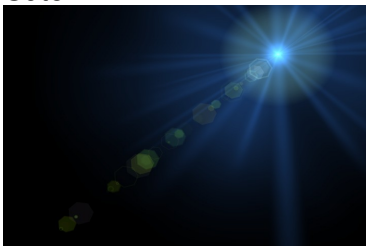
- Spikey



- SmallBlue

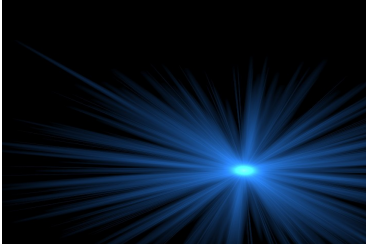


- Cute

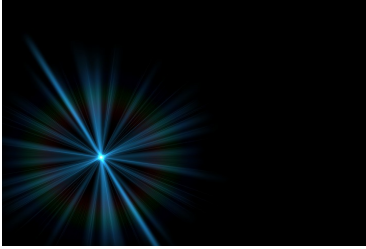


## 5 SpeedSix.LensFlare (Discreet Lenz Raptor)

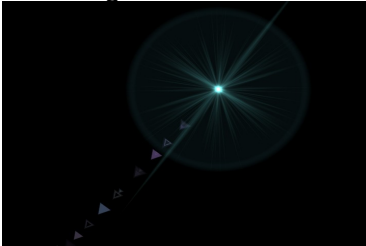
- SoftBlue



- StarBlue



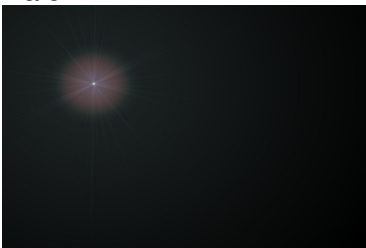
- BlueRing



- PurpleHaze

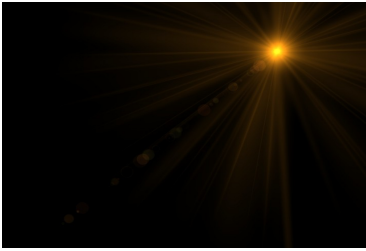


- Halo

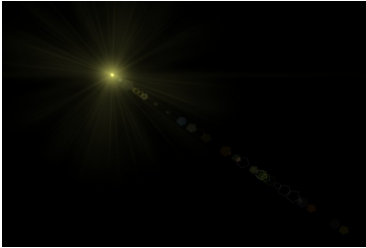


### Warm

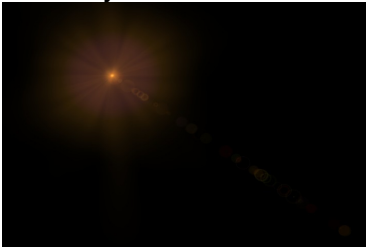
- Blast



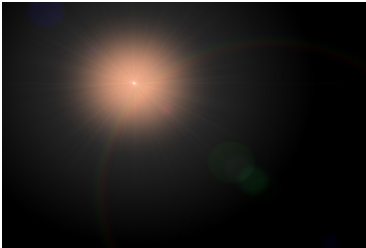
• Burst



• Smoothy



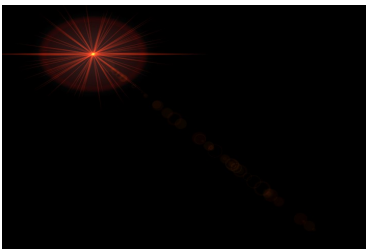
• Dawn



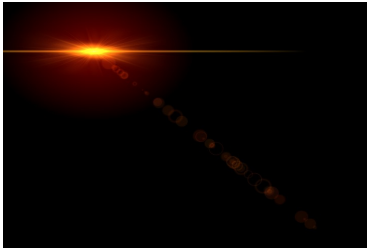
• Golden



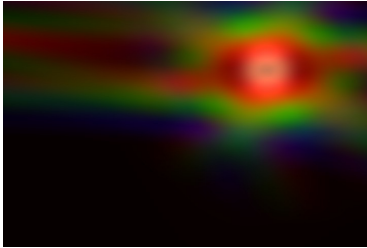
• RedDwarf



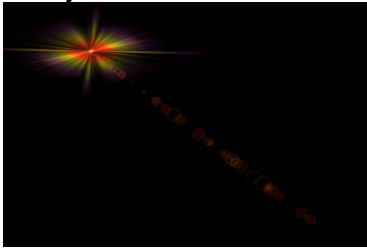
• Hot



- RainbowStar

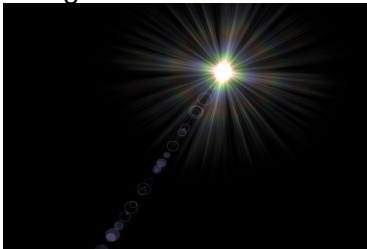


- Pretty

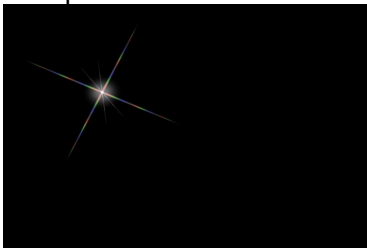


#### More Presets

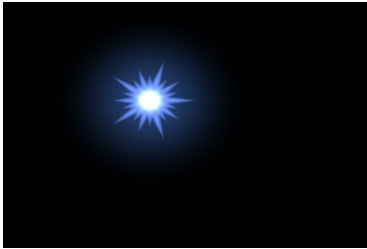
- Halogen



- Sharp



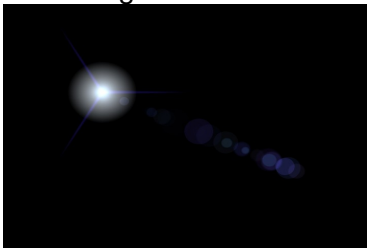
- Graphic



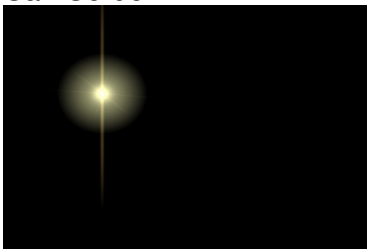
• SmallWhite



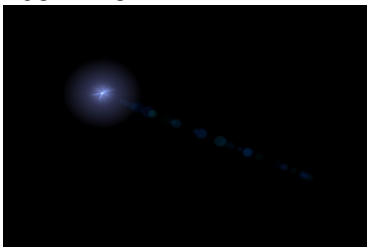
• SearchLight



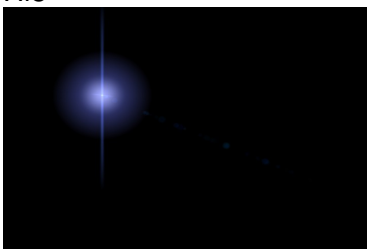
• CamCorder



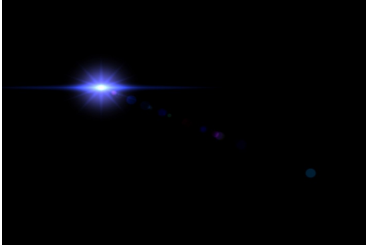
• 105Prime



• Hi8



• SpotLight



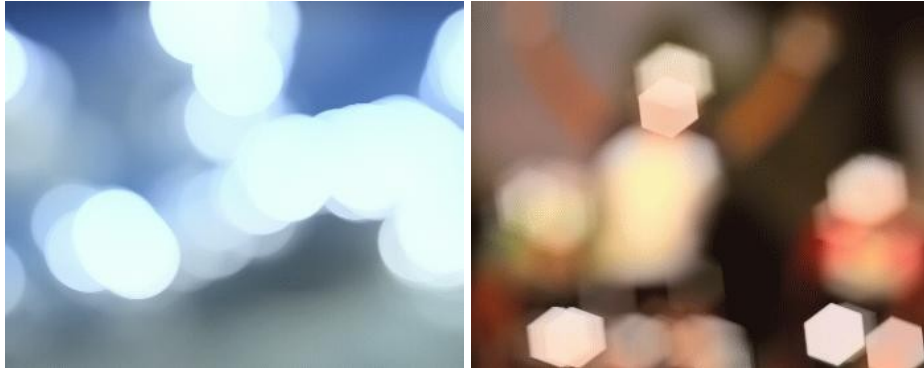


## 6 SpeedSix.RackDefocus (Discreet Lenz Raptor)

---

### PURPOSE

More realistic defocus effect with bright source and aperture shape simulation.



### INPUT CLIPS

**1: Input** : Clip to Defocus

**2: Spread** : An image, or just a section of it, can be used as the template rather than a geometric shape.

---

### *RackDefocus* CONTROL PAGE

**Shape** (List Box **Options:** Circle | Ellipse | Polygon | Image, **Default:** Circle)

**Circle:** the aperture is a circle.

**Ellipse:** the aperture is an ellipse.

**Polygon:** the aperture is polygonal.

**Image:** get point spread function from centre of an image.

**Match Radii** (Checkbox **Default:** On)

Lock X and Y radii together.

**RadiusX** (Number **Min:** 0.000, **Max:** 0.300, **Default:** 0.030)

Controls the radius of a circle (% image height), X radius of an ellipse or polygonal shape.

**RadiusY** (Number **Min:** 0.000, **Max:** 0.300, **Default:** 0.030)

Controls the Y radius of an ellipse or polygonal shape. In % image height.

**Centre** (Position **Default:** 0.5,0.5)

Centre of region of PSF image to use

**RadiusPSF1** (Number **Min:** 0.001, **Max:** 1.0, **Default:** 0.1)

Radius 1 of area to use in PSF image. Image will be faded out between Radius 1 and 2

**RadiusPSF2** (Number **Min:** 0.001, **Max:** 1.0, **Default:** 0.15)

Radius 2 of area to use in PSF image. Image will be faded out between Radius 1 and 2

**Bright Emphasis** (Number **Min:** 1.0, **Max:** 100.0, **Default:** 50.0)

Degree to which bright areas are emphasised.

**Rotation** (Number **Min:** -1000.0, **Max:** 1000.0, **Default:** 0.0)

Rotation of a polygonal aperture.

**Sides** (Number **Min:** 3, **Max:** 20, **Default:** 6)

Number of sides of a polygonal aperture.