G'MIC 3.4.0 : Image Processing in Its Prime!

David Tschumperlé, Garry Osgood

G'MIC *3.4.0* : Image Processing in Its Prime!

By David Tschumperlé and Garry Osgood.

A new version **3.4.0** of **G'MIC** (*GREYC's Magic for Image Computing*) has just been released. With this new release, we celebrate the project's **16th anniversary**!

On this occasion, we summarize the recent features added to our open-source framework for digital image processing, following our previous news on this subject (published in May 2023).

Table of Contents

_ **Note**: Click on the images of the article, to get a full-resolution version, or to play a video when the images contain the icon `🎞_

1. *G'MIC* in a Nutshell

G'MIC is an open-source framework for digital image manipulation and processing, developed in the *IMAGE* team at the *GREYC* research laboratory in Caen (*UMR CNRS* 6072).

It defines various user interfaces for applying a wide range of algorithms to images and signals. The core of the project is the *G'MIC* scripting language interpreter specifically designed to ease the prototyping and implementation of new image processing algorithms and operators. Users can harness operators from the several hundred already implemented, or write their own custom operators. These can then be accessed through the several user interfaces that the project offers.

The most popular G'MIC interfaces are:

1. `gmic` , a command-line tool, and a useful complement to *ImageMagick* or *GraphicsMagick* for processing, generating, or analyzing images from a *shell*;
2. *G'MIC Online* a web service where many *G'MIC* filters can be applied on images, directly from a web browser;
3. *G'MIC-Qt* a popular plugin supported by many digital image editing applications such as *GIMP*, *Krita*, *Paint.NET*, and *Photoshop* (thanks to its availability as a *8bf plugin*).

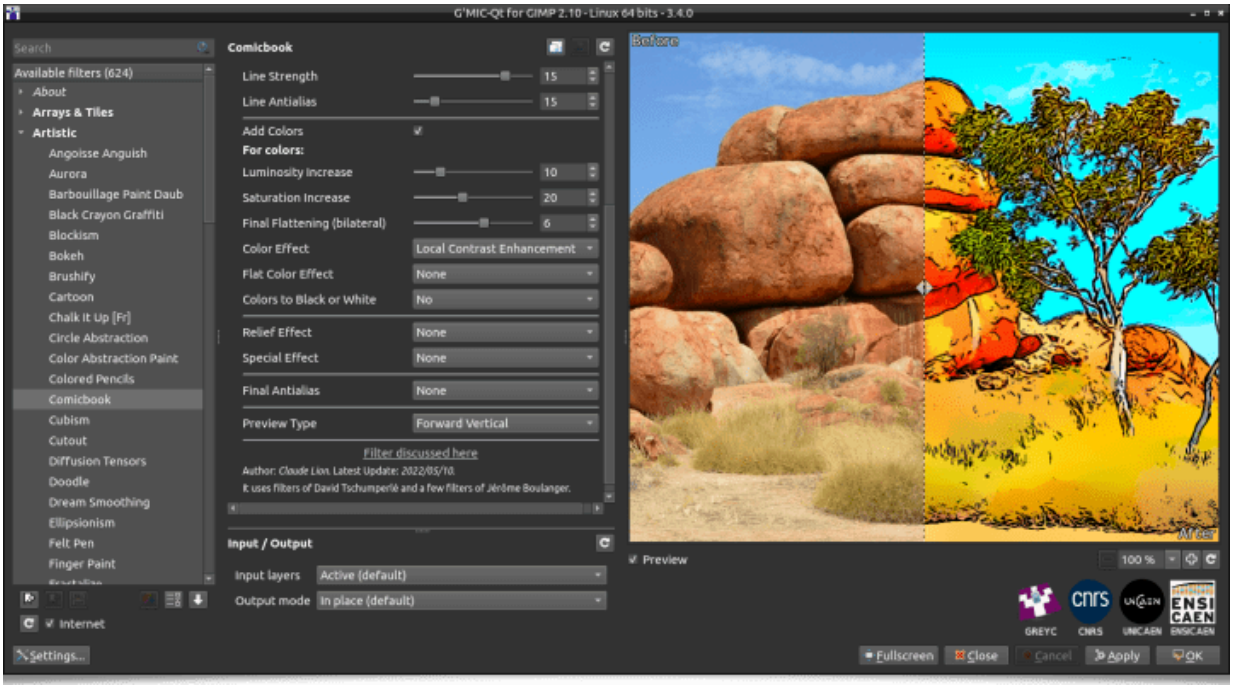


*Fig. 1.1. Preview of the G'MIC-Qt plugin version **3.4.0**, running in GIMP 2.10, with the Comicbook filter selected.*

2. What's New in Version 3.4.0?

Version **3.4.0** of *G'MIC* focuses on *stability* and *long-term support*: After more than 15 years of continuous development, *G'MIC's* core elements have been extensively put through their paces. We now strive for stabilizing the key *Application Programmer's Interfaces (APIs):* `libgmic` , to integrate *G'MIC* features into *C* or *C++* code, *G'MIC-Qt* plugin to port the plugin to new hosts, and the syntax of the *G'MIC* script language itself, including the freezing of the parameter settings of the *G'MIC* commands already available.

Thus, all versions of the *3.4.x* branch will be dedicated to bug fixing and developing features that do not require modification of the language core, such as developing new filters or image processing operators. Attention will be paid to ensuring maximum backward compatibility among *3.4.x* releases. In this sense, this version *3.4.0* represents a long term milestone release in the project's history.

That said, in the run-up from versions *3.3* to *3.4*, there have been notable improvements to several user interfaces:

- ** *G'MIC Online* ** (*G'MICol*): This web service allows users to apply *G'MIC* filters to their uploaded images through their web browsers. It has existed for several years, largely unchanged from its original form, while interactive web techniques have advanced. In 2024, the *DDA* (*Application Development and Deployment*) team at the *GREYC* laboratory undertook a complete overhaul of *G'MIC Online*. The web service now sports a new look, employs significant user interface improvements, with more intuitive widgets, an enhanced preview window, a choice between light and dark themes, revamped handling of filter favorites and better integration of the web service with the *G'MIC* core, such that web users can now access recently developed filters and image processing pipelines.

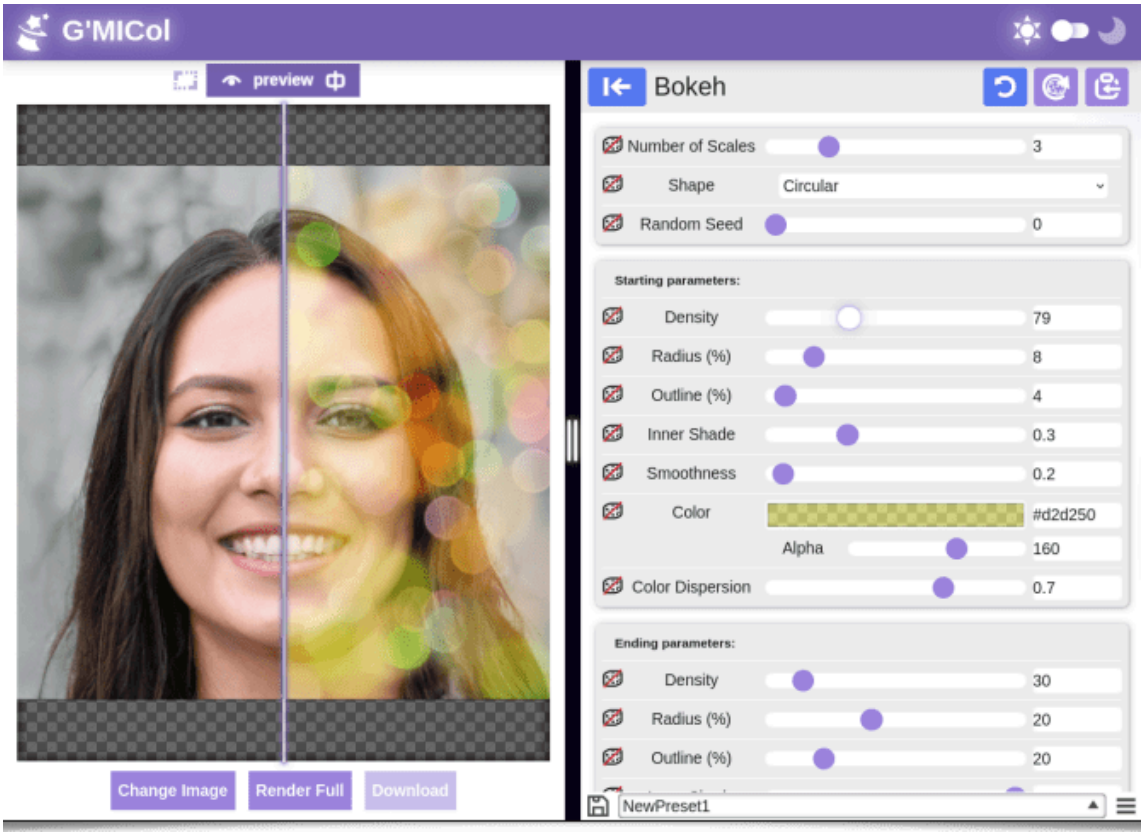


Fig.2.1. Preview of the revamped G'MIC Online web service released with version 3.4.0 (here with the light theme).

Visit *G'MIC Online* and give it a real-world test 😱! And don't hesitate to report any issues. In any case, a big thank you to our colleagues in the *DDA* team for this noteworthy overhaul!

- ** *G'MIC-Qt* Plugin**: Over the past year, the plugin has acquired more than **40 new image processing filters**, detailed below. Many employ the new *Random Parameters* button, which generates arbitrary filter settings, furnishing rapid previews that can be quickly evaluated for practical or aesthetic worth.

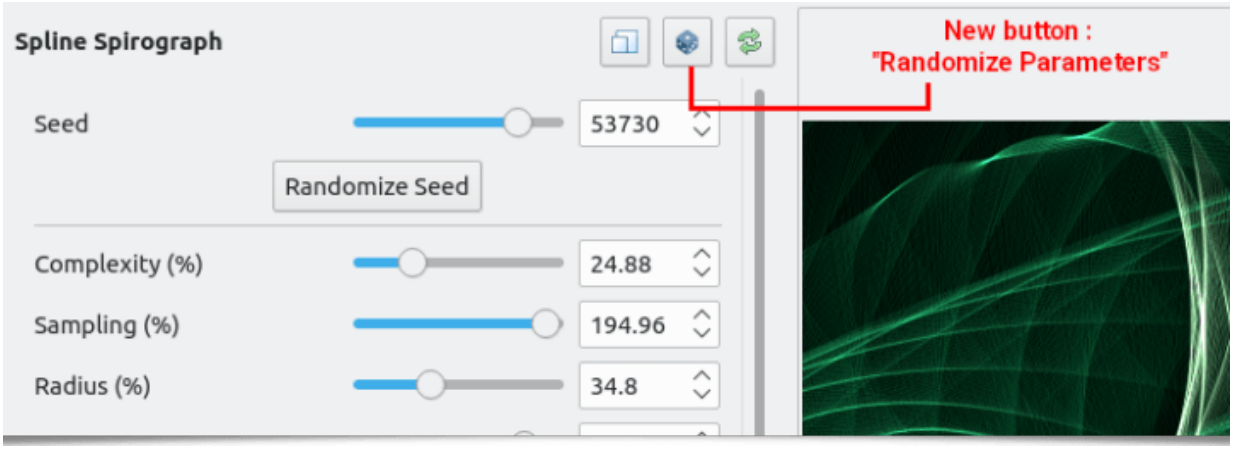


Fig.2.2. G'MIC-Qt Plugin: The new Random Parameters button assigns random values to the selected filter's parameters.

- **`gmic` CLI Tool**: All filters created by the developer community are now integrated within the command-line tool executable. There are no longer discrepancies in the number of default filters available between the *G'MIC-Qt* plugin and the `gmic` CLI tool. Previously, users needed to force filter updates with `$ gmic update` to align environments.

`gmic` now offers over **4000 functions** to process your images from the command line. Notable efforts have been made to improve the integrated image viewer available through the `display` command, which has been completely re-tooled to allow viewing different image types within the same interface.

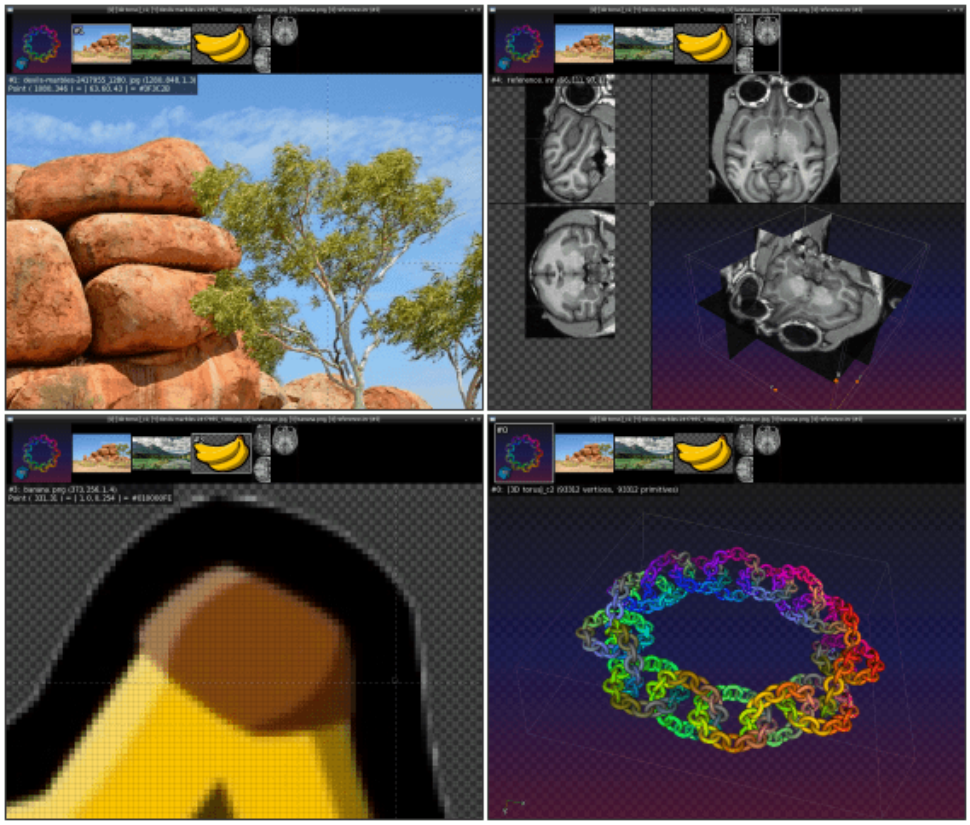


Fig.2.3. `gmic` CLI Tool: The G'MIC image viewer has been completely re-implemented and allows viewing different image types within the same interface.

This is also the case for the mathematical function or 1D signal viewer (command `plot`), which has been revamped:

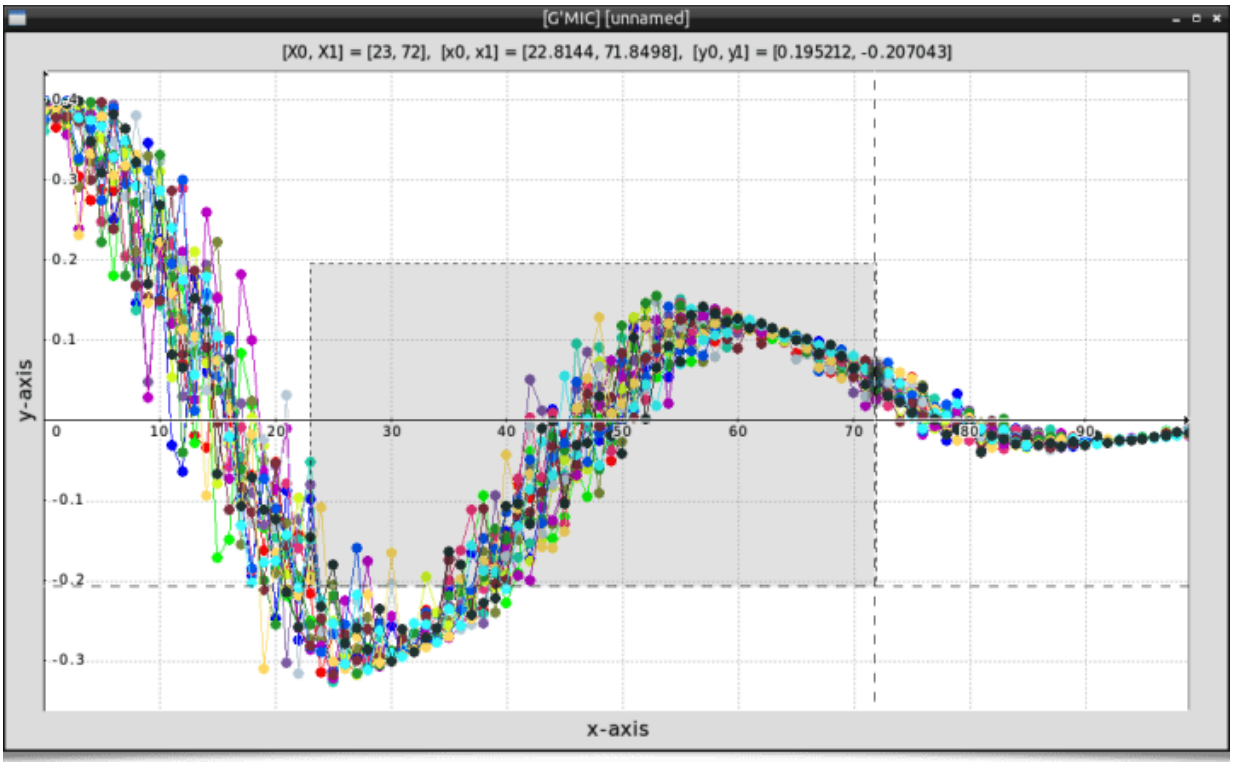


Fig.2.4. `gmic` *CLI Tool: The G'MIC viewer for mathematical functions or 1D signals has also been re-implemented.*

This makes `gmic` a handy tool, even if we limit ourselves to viewing images or signals on the command line.

- **Other Improvements:**
  - The integrated mathematical expression evaluator has evolved significantly: it is one of the essential bricks of the *G'MIC* language, as it is responsible for evaluating the results of the mathematical operations needed in scripts (needless to say, calculations are done all the time in image processing). This evaluator has been enriched with many new functions, especially for the calculation and processing of vectors and matrices, and native manipulation of heaps.
  - Many native (*C++*) functions formerly in the shared binary object library have now been entirely rewritten in the *G'MIC* language. This facilitates their maintenance and further development, as future work can be scripted for immediate effect, instead of being subject to compilation and linking. Situated at the script level, such modifications become immediately available to users, as distribution of binaries will no longer be required.
  - The commands for loading/saving 3D meshes in `.obj` (*Wavefront*) format have been improved, and new commands for processing/creating 3D meshes have emerged (e.g., to visualize the normal vectors at the vertices of a mesh, as illustrated in the video below). And, no, this is not a rotten apple:

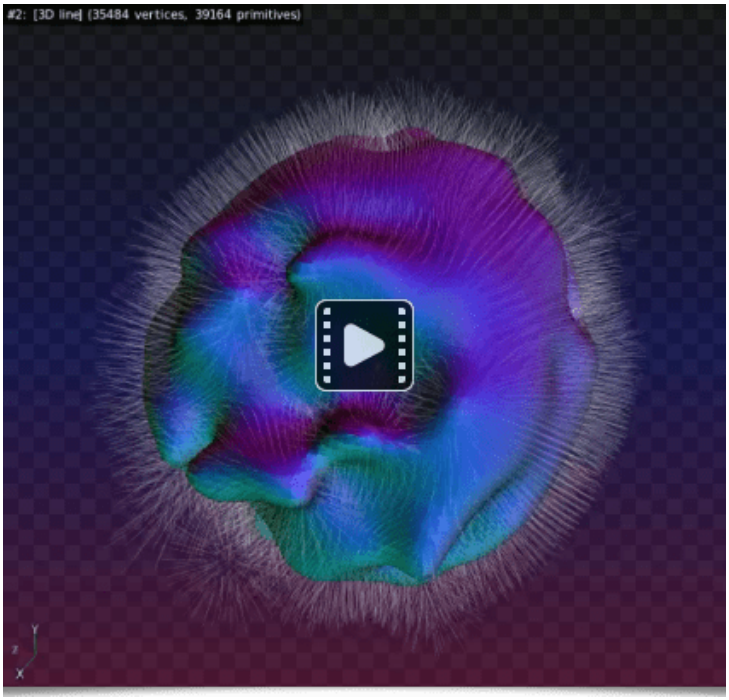


Fig.2.5. 3D mesh management within G'MIC has been enriched with new commands.

That's all for the general improvements of the various interfaces offered by the framework. Let's now move on to the details of the new filters and image processing features that have appeared in *G'MIC* over the past twelve months.

3. Details of the New Filters

In this extensive section, we categorize and describe the recent filters by type of use: *Photo Retouching*, *Distortion Filters*, *Degradation Filters*, *Rendering and Texture Filters*, and finally *Artistic Effects*.

3.1. Photo Retouching

Three interesting filters have appeared in the *G'MIC-Qt* plugin to help photographers retouch their digital shots.

First, the **Colors / Mixer [Generic]** filter, a color channel mixing filter that offers the possibility to choose from no less than 16 different color spaces/representations for mixing (*CMY*, *CMYK*, *HCY*, *HSI*, *HSL*, *HSV*, *Jzazbz*, *Lab*, *Lch*, *OKlab*, *RGB*, *RYB*, *XYZ*, *YCbCr*, *YIQ*, *and YUV*). This constitutes a good alternative to traditional contrast or color enhancement tools, to retouch photos that might be a bit dull.

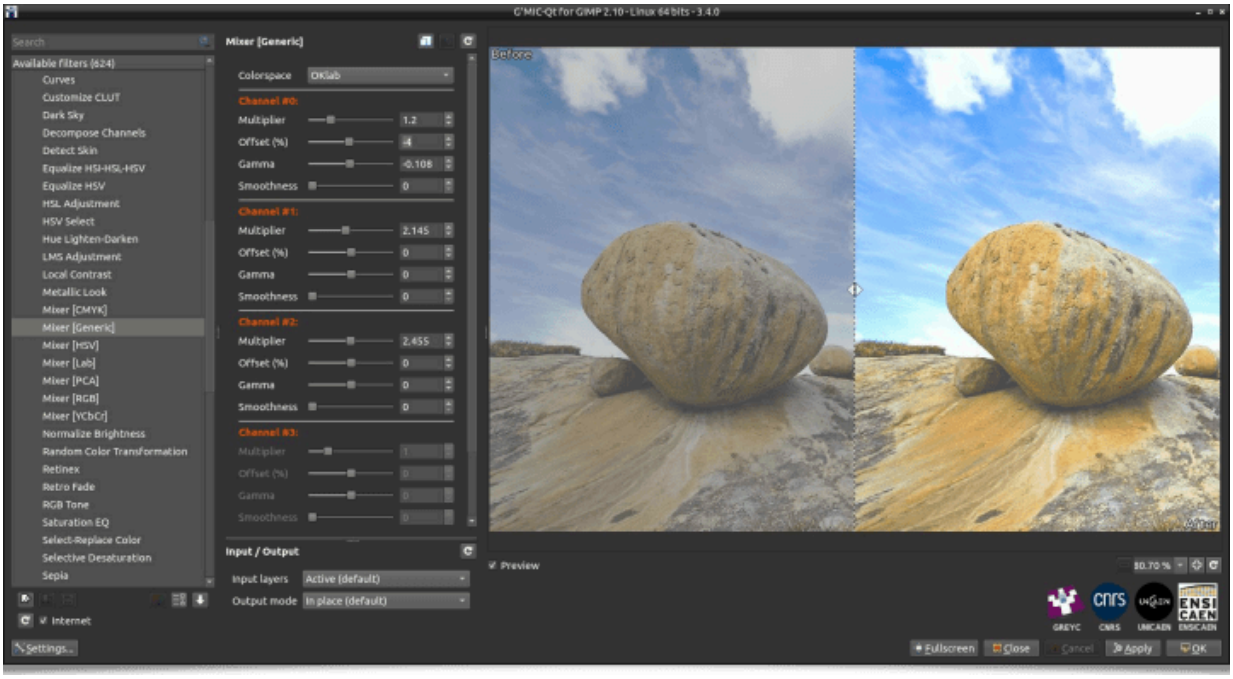


*Fig.3.1.1. The **Colors / Mixer [Generic]** filter enriches the already available arsenal of filters for contrast and color enhancement.*

Let's also talk about the **Details / Sharpen [Alpha]** filter, which, as its name suggests, allows for fine detail enhancement in photographs. It minimizes the appearance of edge halos, a common artifact arising from many detail enhancement filters. The **Details/Sharpen [Alpha]** filter is based on an original technique of pyramidal decomposition of the image obtained from an *Alpha blending* operator.

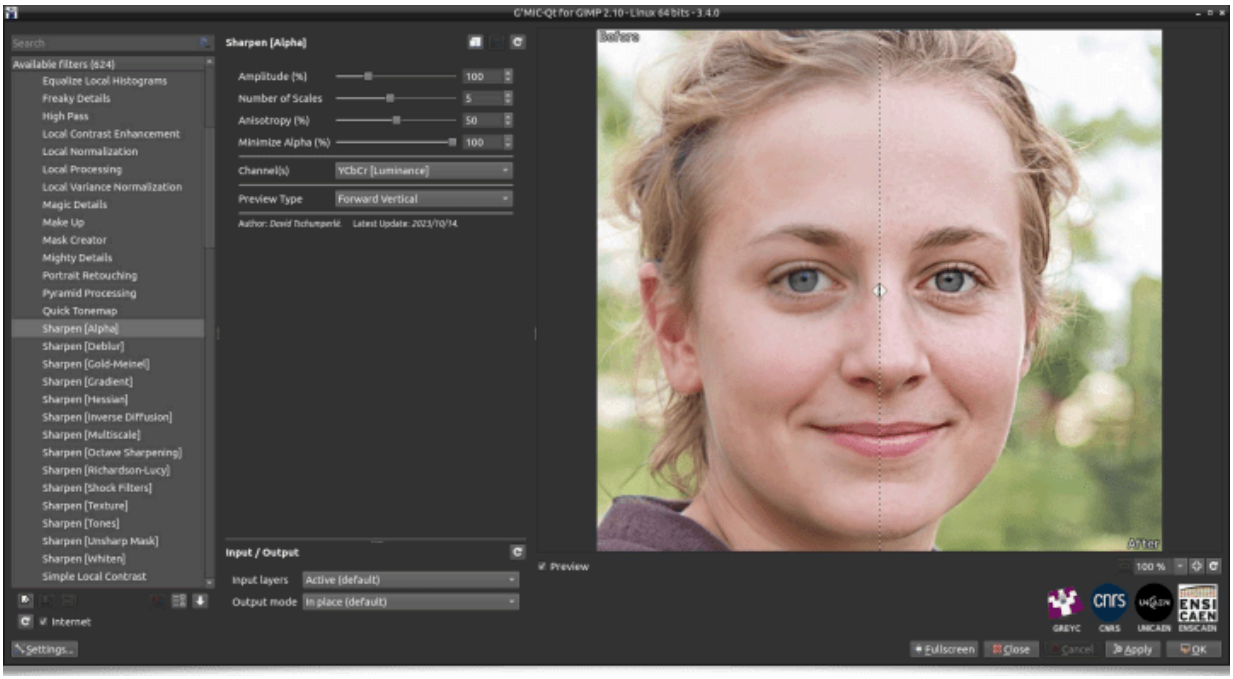


*Fig.3.1.2. The **Details / Sharpen [Alpha]** filter allows for fine detail enhancement in photographs, minimizing the appearance of halos.*

*Fig.3.1.3. Before/After details illustrating the outcome of the **Details / Sharpen [Alpha]** filter.*

Finally, the **Layers / Spatial Blend Multi-Layers** filter merges into a single image multiple shots from the same viewpoint, while harnessing a spatial linear gradient between the various views. For example, one can photograph the same scene over day and night time intervals, then employ this spatial fusion filter to create time-transitional images like the one illustrated below:

*Fig.3.1.4. The **Layers / Spatial Blend Multi-Layers** filter allows for merging multiple photographs with an adjustable spatial linear gradient.*

(**Credits**: the images used in the figure above are from the video *Stunning New York City skyline timelapse: Day to night* by the YouTube channel *Rumble Viral*.)

The following video illustrates the complete process, using the *G'MIC-Qt* plugin under *GIMP 2.10*:

3.2. Distortion Filters

Let's now move on to a set of new effects available to distort your images in various ways.

First, the **Deformations / Distort [RBF]** filter, which deforms an image based on adjustable key points. A *Radial Basis Function* interpolation function alters distances non-linearly, applying contractions here, expansions there, all in a user-specifiable way. Mathematically inclined users can specify their own radial basis functions, placing few limits on the kinds of possible distortions.

*Fig.3.2.1. The **Deformations / Distort [RBF]** filter allows for various distortions based on RBFs. Here, specifying the radial basis function phi(r) = log(0.1+r).*

Next, let's mention the arrival of an entirely new category **Map Projection**, which now contains fourteen filters for maps initially in the form of *equirectangular projection* (cylindrical equidistant maps). These may be further converted into other types of projections. This series of filters has been contributed by *Kristian Järventaus*, a member of the *Cartographers Guild forum*, a forum specialized in cartography.

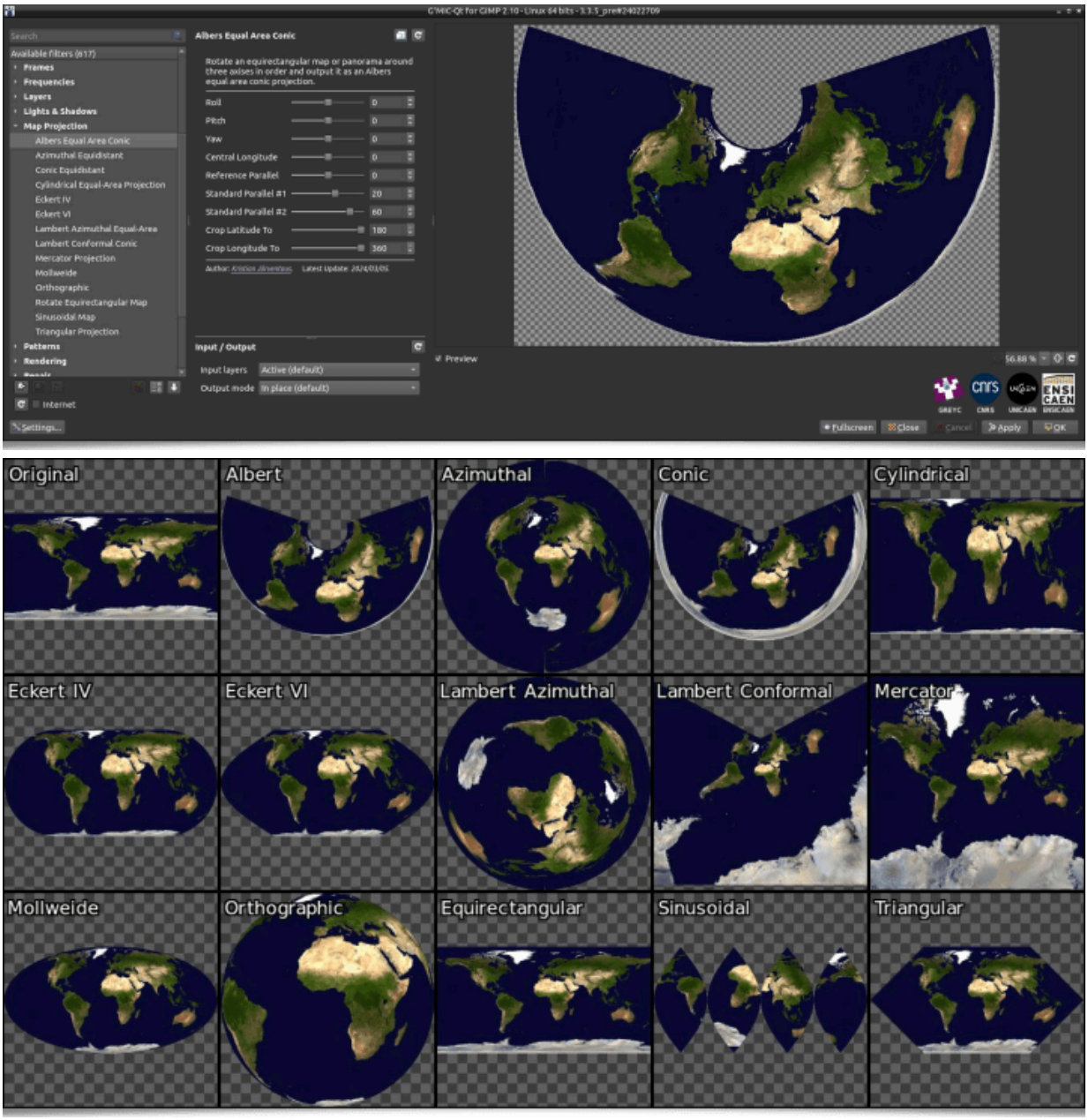


*Fig.3.2.2. The new **Map Projection** filters offers several geographical map projection algorithms.*

We also note the arrival of the **Deformations / Square to Circle [alt]** filter, which transforms any rectangle (or square) centered in an image into an ellipse (or circle), and vice versa. A very specialized filter, with — at first glance — obviously few apparent applications, yet when you need it, you'll be glad to have it at hand! For example, we used it below to transform a round painting frame into a square frame:

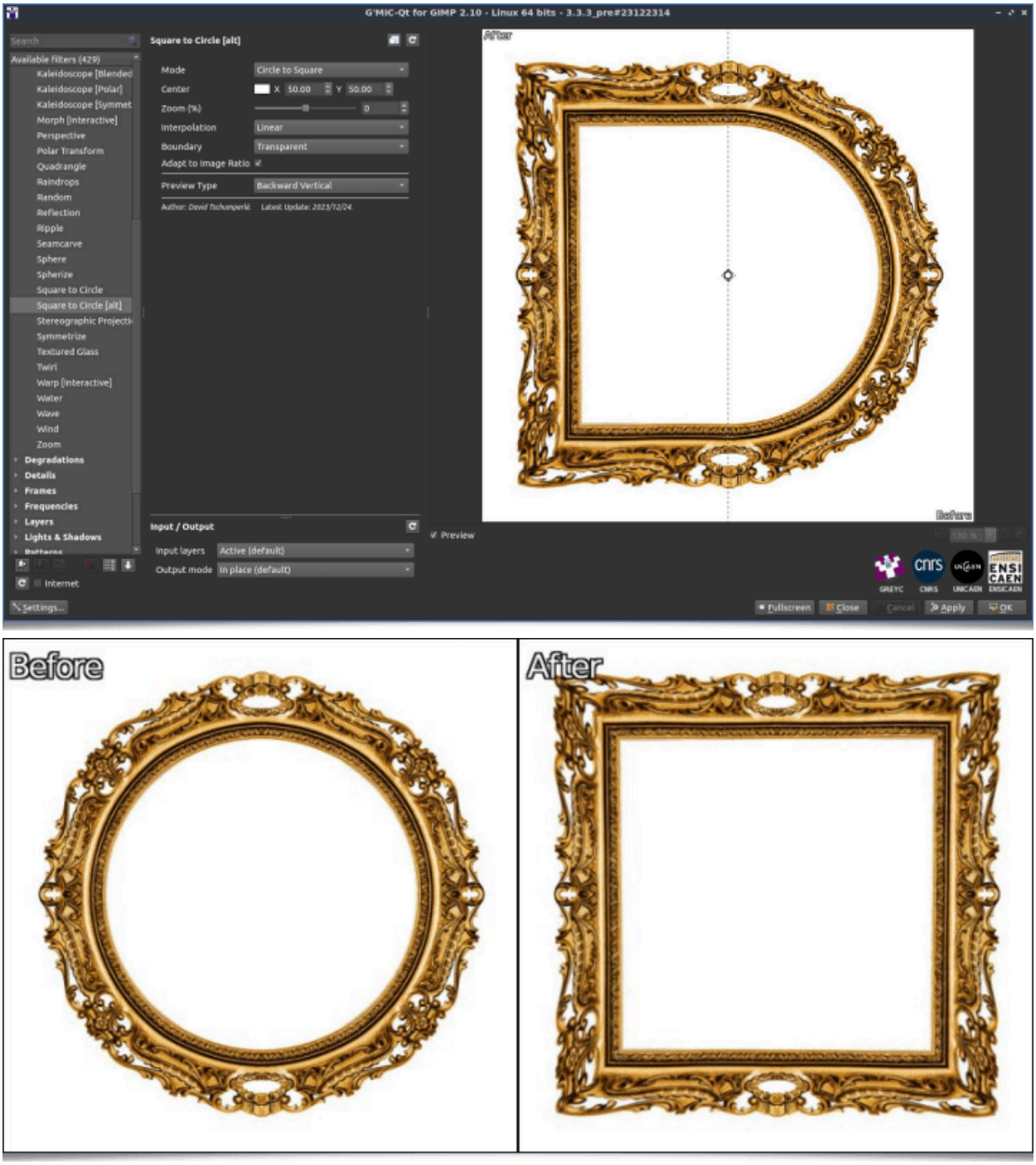


*Fig.3.2.3. The **Deformations / Square to Circle [alt]** filter allows converting square or rectangular objects into round or elliptical objects, and vice versa.*

Finally, let's conclude this review of the new image deformation filters with the **Deformations / Poincaré Disk** filter, which generates *Poincaré disks*, a family of geometric figures based on hyperbolic geometries, as illustrated in the following figure:

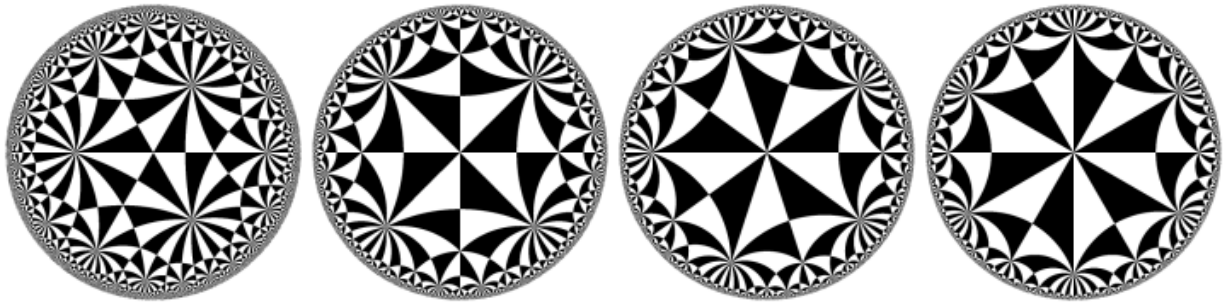


*Fig.3.2.4. Some examples of Poincaré disks, generated by the **Deformations / Poincaré Disk** filter.*

But where it becomes interesting is that this filter also allows you to deform images by projecting them onto these very particular geometries:

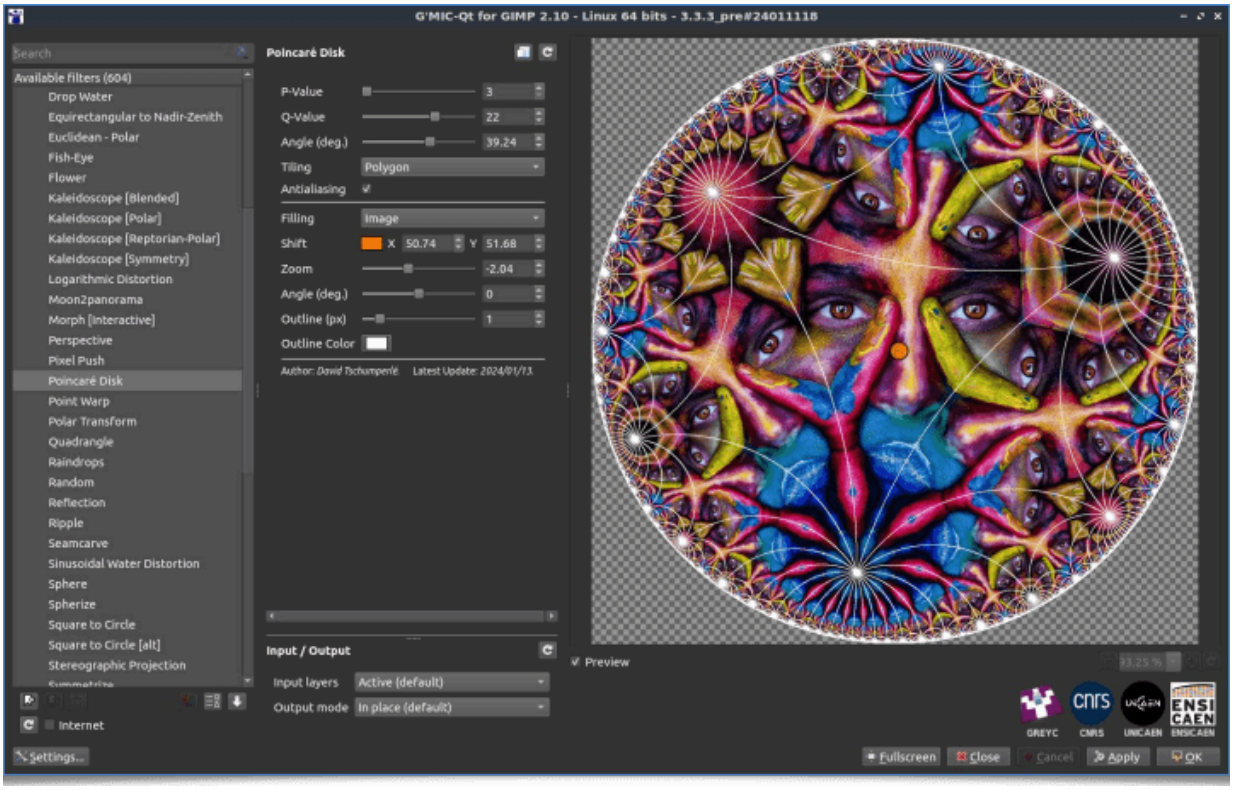
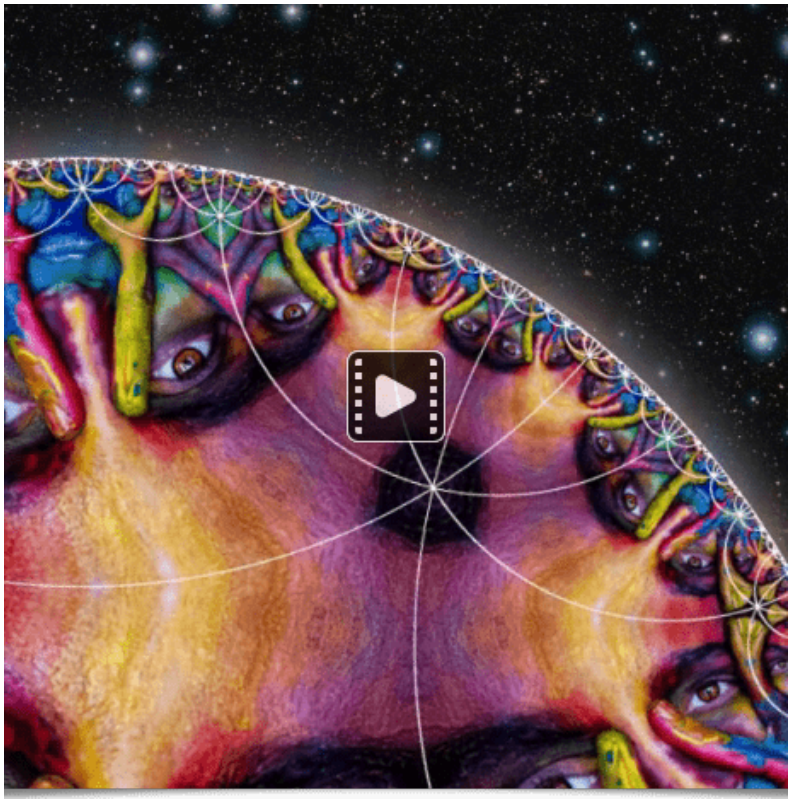


Fig.3.2.5. Projection of an image onto a Poincaré disk.

For example, we used this filter, with slight modifications, to generate the short animation, *"Flyover of the Poincaré Planet"*, which illustrates the astonishing fractal properties of these geometric oddities:



3.3. Degradation Filters

Sometimes, one seeks to deliberately *degrade* images, either to simulate a real phenomenon, such as motion blur, sensor noise and the like, or in the pursuit of a purely aesthetic effect (*Glitch Art*). For these purposes, the following new effects have been added to *G'MIC*:

- The **Rendering / CRT Scanlines** and **Degradations / CRT Phosphors** filters mimic the image display of *cathode-ray tubes (CRT)*, by simulating two characteristic effects of CRT displays, namely the *Scanline effect* and *phosphor excitation*. These two filters were created in collaboration with *Romain Hérault*, a new contributor who recently joined the *GREYC* laboratory.

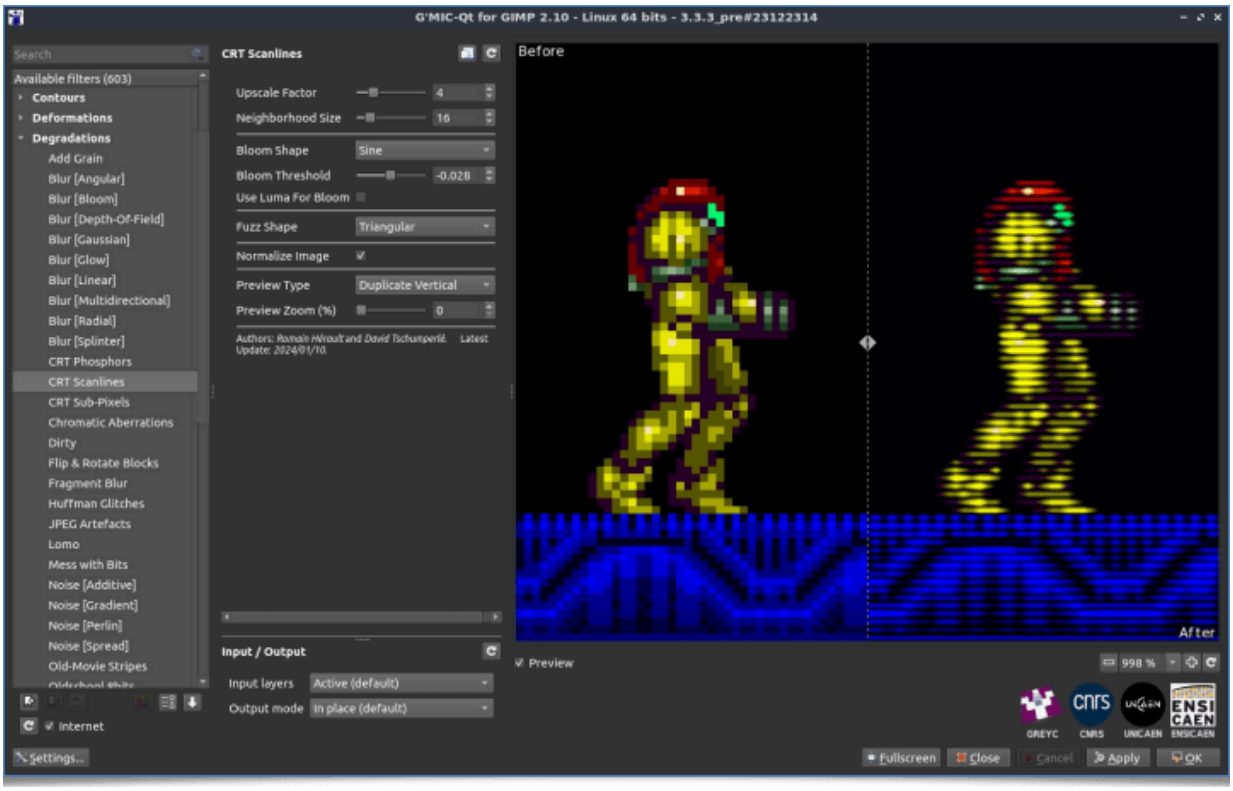


Fig.3.3.1. The ***Rendering / CRT Scanlines*** *filter imitates the typical scanline effect of CRT displays.*

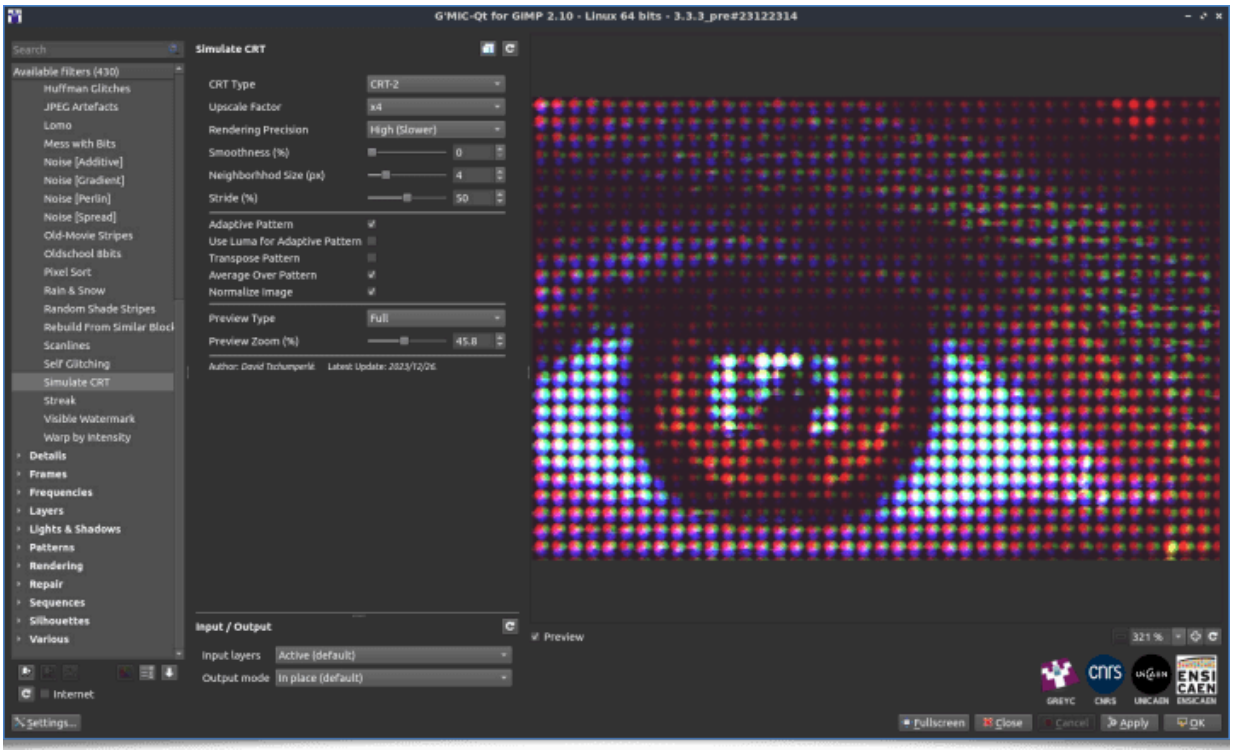


Fig.3.3.2. The ***Degradations / CRT Phosphors*** *filter simulates the phosphor display technique of CRT displays.*

- The **Degradations / Blur [Motion]** filter emulates path-based motion blur, with user-adjustable, spline-based trajectories which may be directly altered through key points in the *G'MIC-Qt* preview window:

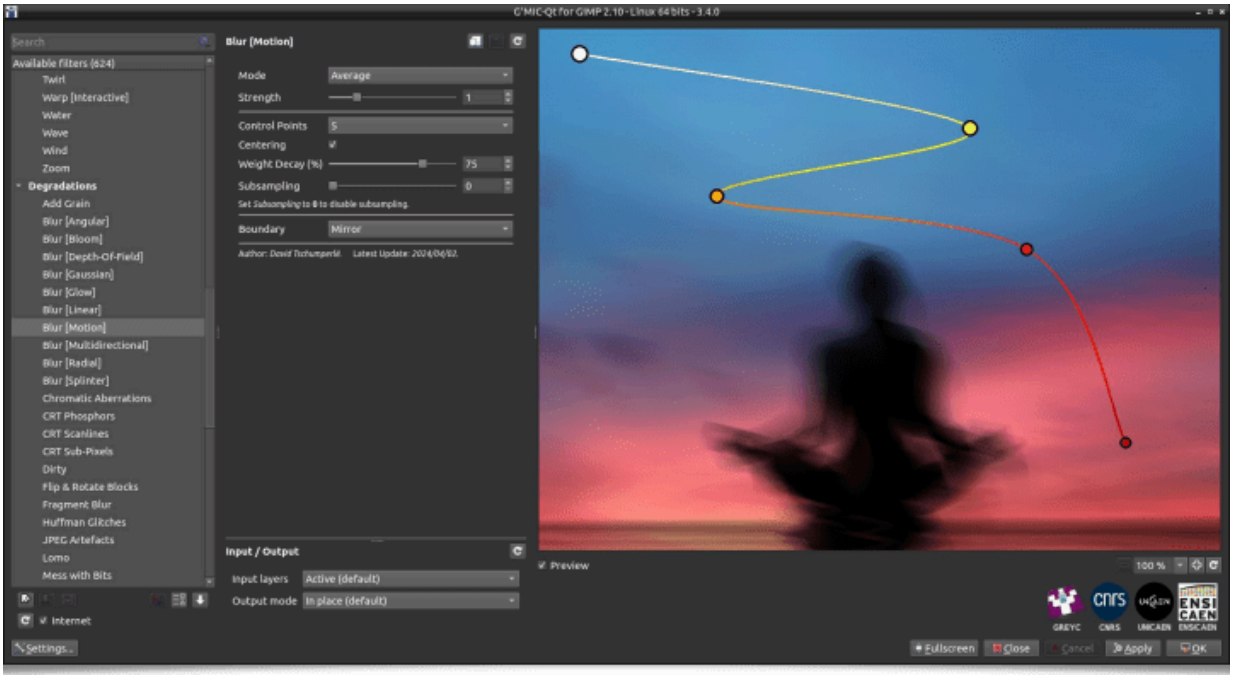


*Fig.3.3.3. The **Degradations / Blur [Motion]** filter simulates motion blur.*

- The **Degradations / Sloppy Mess** filter is intended for Glitch Art enthusiasts. It's one of the first filters by a new contributor, Prawnsushi, who recently took an interest in the *G'MIC* language for filter creation. This effect offers wide-ranging parameters to produce a panoply of disjoint and disassociated outputs.

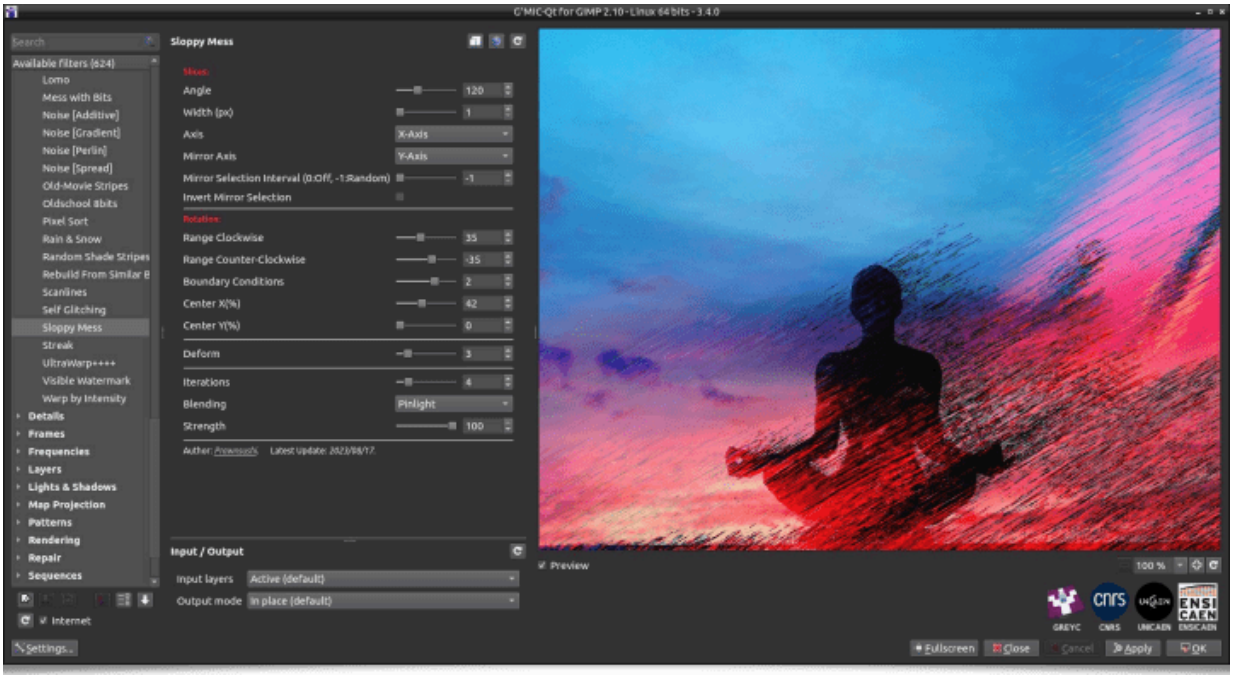


*Fig.3.3.4. The **Degradations / Sloppy Mess** filter deliberately creates artistic digital artifacts on your images.*

3.4. Rendering and Texture Filters

Rendering filters, by and large, do not require input images; they summon new imagery by algorithmic means through *procedural generation*.

- The **Rendering / Underwoods** filter, another creation of *Prawnsushi*, is nothing less than an **underwood generator**! One might say it's not very useful, and indeed it's not a filter that image processors,

photographers, or illustrators will incorporate into their daily work.

Yet this filter illustrates the whole philosophy of the *G'MIC* project: To produce software that fosters **algorithmic creativity**. *G'MIC* facilitates the construction of **all kinds of filters** (useful or less useful), and allows **free sharing** with other *cognoscenti* of this viewpoint. In practice, this filter has the simple merit of its own existence, there for anyone to try, and thereby — possibly! — discover the extraordinary. It uses very little memory resources, perhaps a few dozen bytes, thanks to the compactness of the *G'MIC* language design. And moreover, the results are quite cool! And be assured that it will inevitably serve, one day or another, an artist wanting to synthesize an underwood image in two mouse clicks!

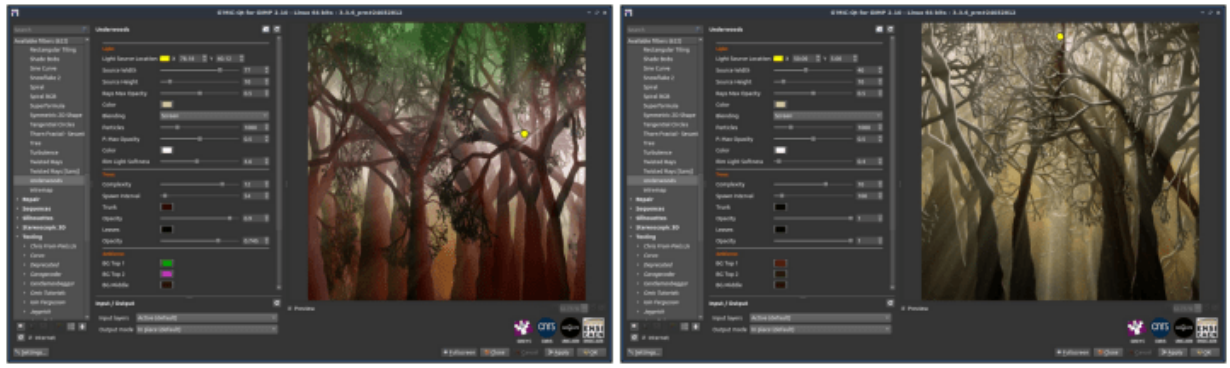


*Fig.3.4.1. Examples of underwood image renderings by the **Rendering / Underwoods** filter.*

- The **Patterns / Reaction-Diffusion** filter synthesizes textures modeled after *reaction-diffusion systems*.

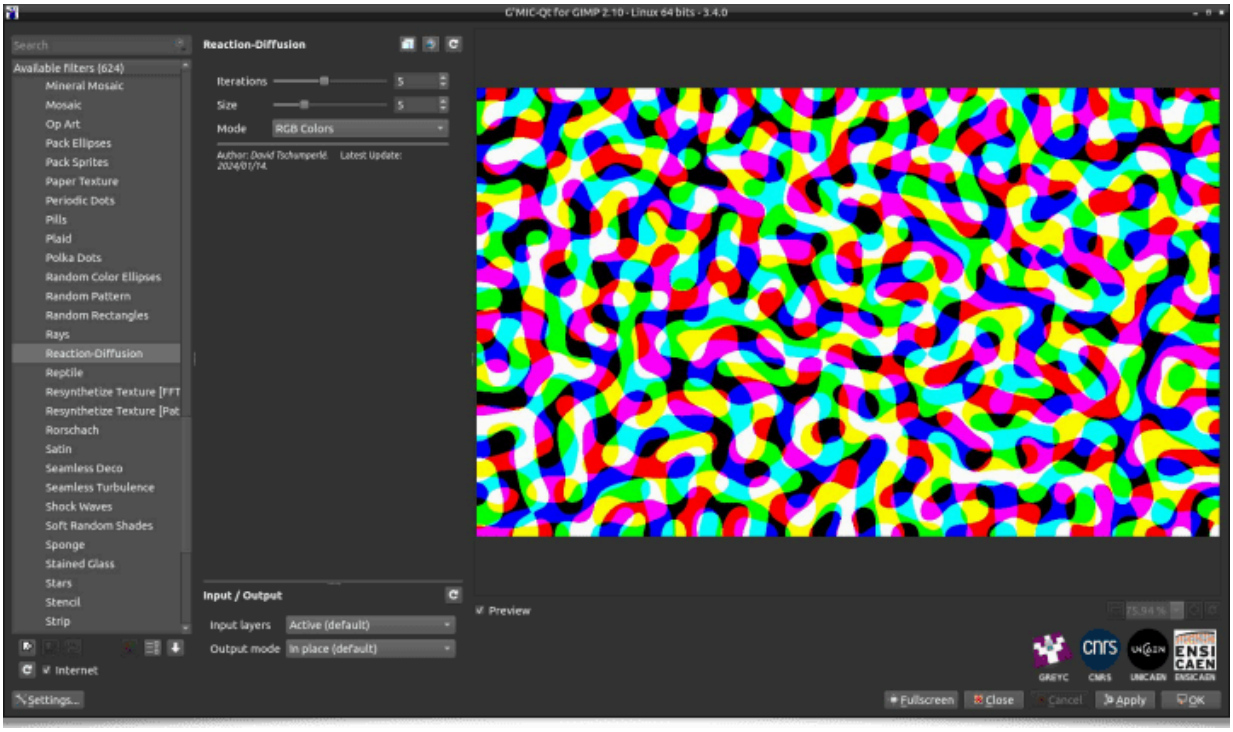
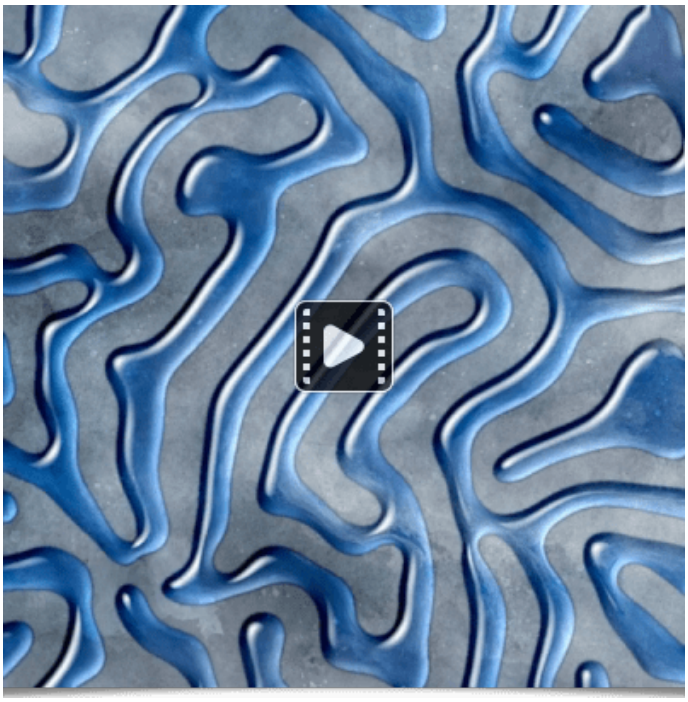
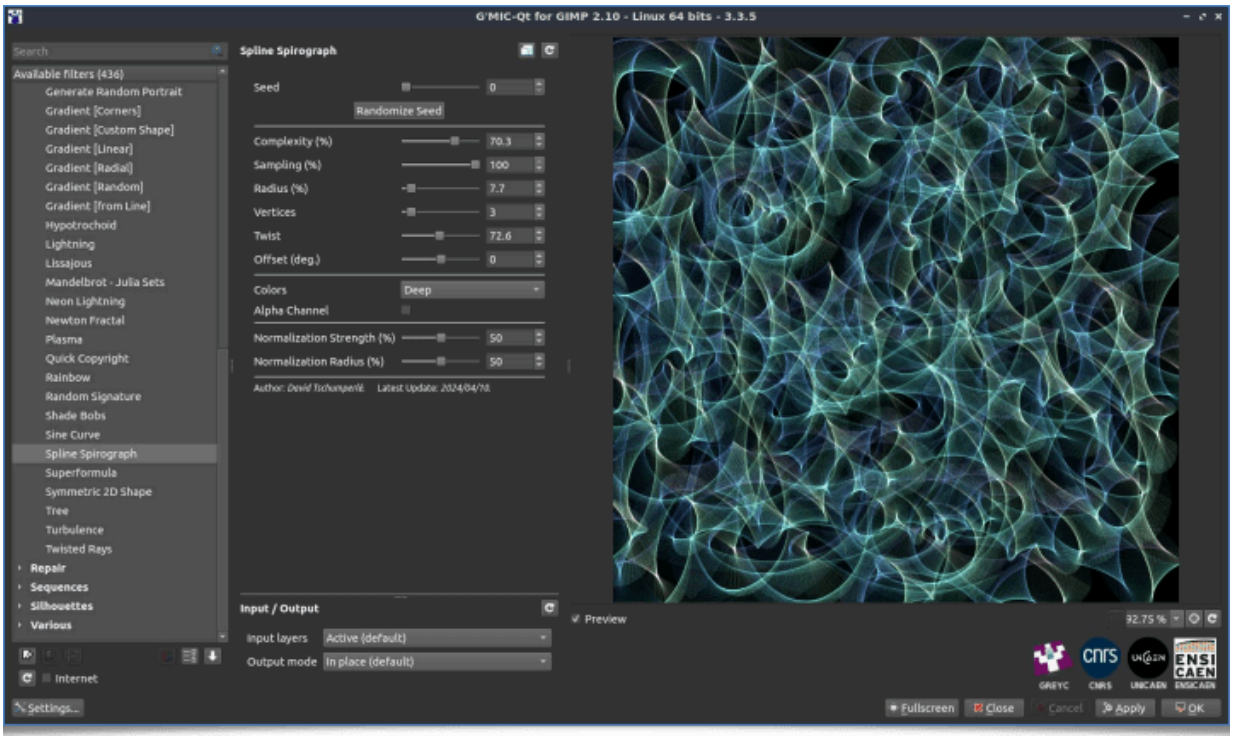


*Fig.3.4.2. The **Patterns / Reaction-Diffusion** filter in the G'MIC-Qt plugin.*

Again, the interest in this type of filter may seem quite limited. However, it's the perfect example of a texture that — in a pipeline of filters of a similar ilk — serves as the basis for generative art. Take a texture created by this filter, apply some of the other effects available in *G'MIC* (e.g., the **Deformations / Drop Water** filter), and you're ready to produce fun animations like the one below:



- The **Rendering / Spline Spirograph** filter is inspired by the *Spirograph* game for generating parametric curves, creating intriguing texture and color effects. The presence of the *Random Parameters* button in the *G'MIC-Qt* plugin is particularly welcome here, as it quickly obtains a varied panorama of diverse results!



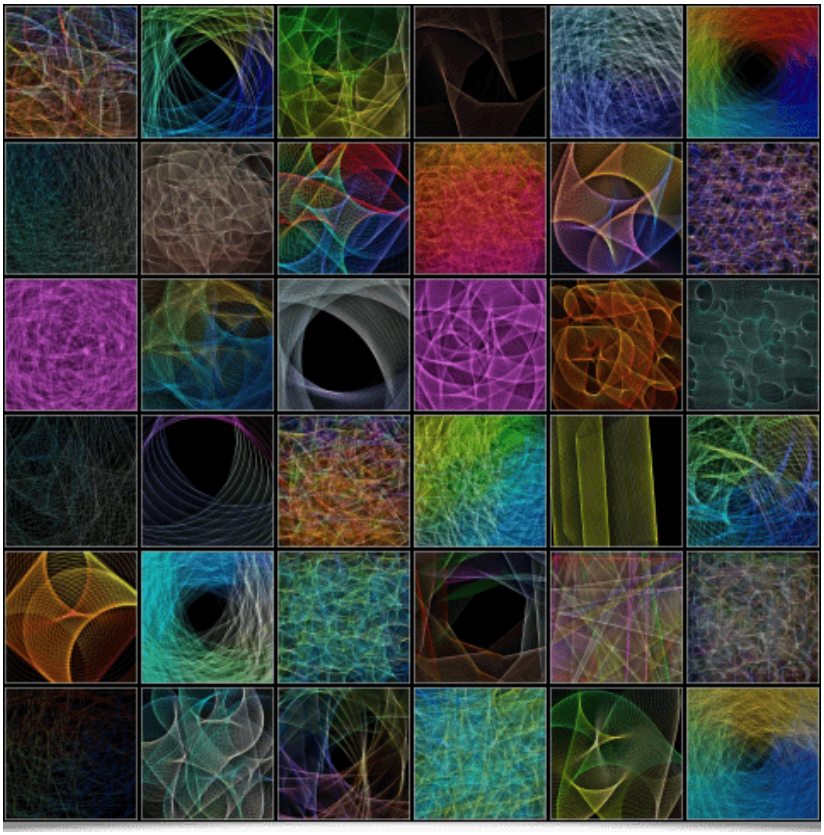
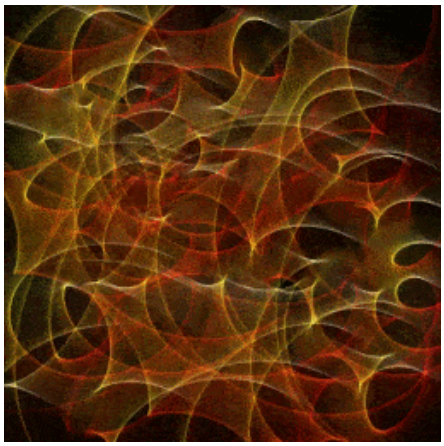


*Fig.3.4.3. The **Rendering / Spline Spirograph** filter and some rendering examples.*

This filter also offers an animated output mode, allowing for the synthesis of short videos like this one:



- The **Rendering / ABN Filigrees** filter also knows how to draw interesting parametric curves, inspired this time by the filigrees found on stock certificates or paper money. This filter pays homage to the *American Bank Note Company*, an engraver of financial paper, active on Wall Street from the 19th century until sixty years ago.

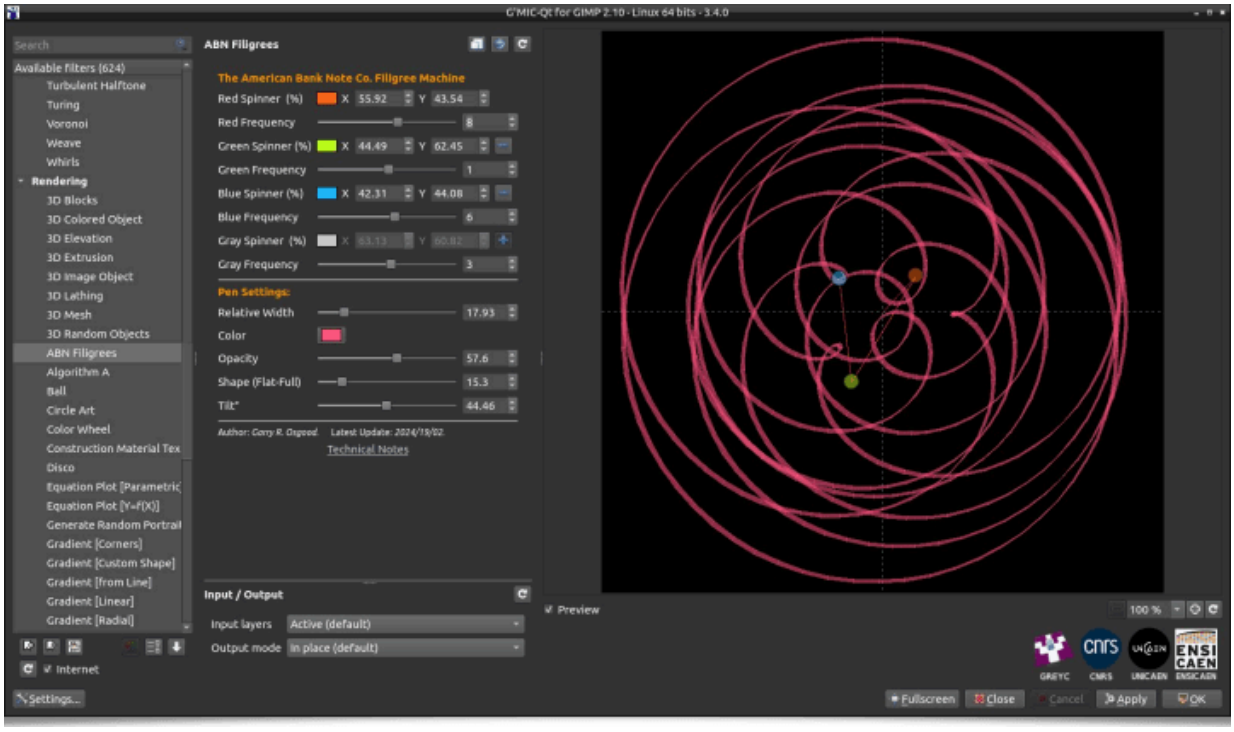


*Fig.3.4.4. The **Rendering / ABN Filigrees** filter.*

- The **Rendering / Random Signature** filter is also an amusing curve generator: it aims to draw random signatures, as illustrated in the video below:

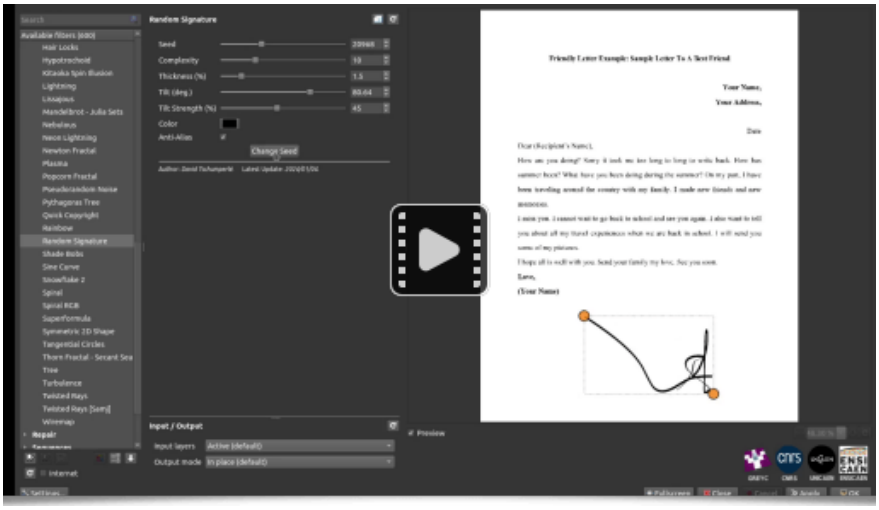


*Fig.3.4.5. The **Rendering / Random Signature** filter creates random signatures.*

Again, a filter whose usefulness cannot be debated: *"It's completely useless, so it's absolutely indispensable!"*.

- And to finish this section, let's mention the **Rendering / Twisted Rays** filter which, as its name suggests, generates a twisted rays effect. The use of this filter will probably remain quite unfathomable in the near term. But, even now, there may be among thousands of *G'MIC* users a creator seeking this precise effect — *and now they can!* ☺

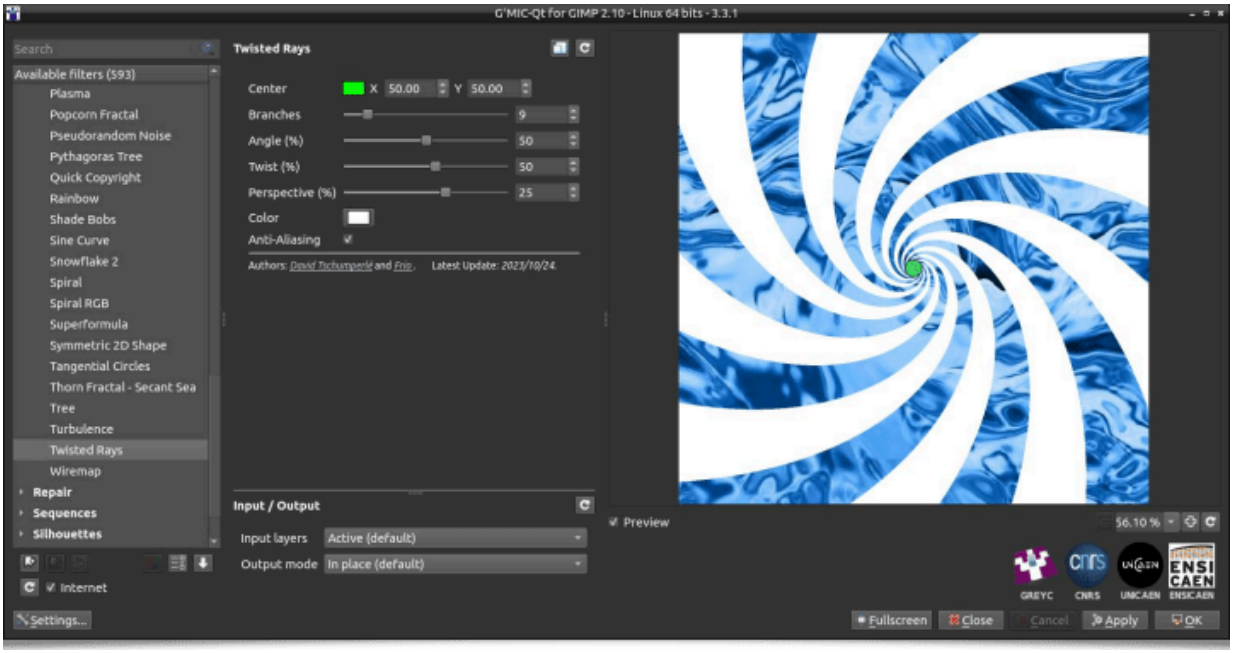


*Fig.3.4.5. The **Rendering / Twisted Rays** filter creates a swirl of rays on your images.*

This filter can, for example, be used to create psychedelic animated effects like this:

3.5. Artistic Effects

To conclude this list of new *G'MIC* filters, here are some miscellaneous filters grouped under the term "Artistic", a term used in image processing software for filters that are hard to categorize...

- The **Artistic / Stringify** filter decomposes an input image into quantized color blobs and connects the points (subsampled) located on the contours of these blobs with color segments. The rendering may resemble the curves obtained with the Spirograph game.

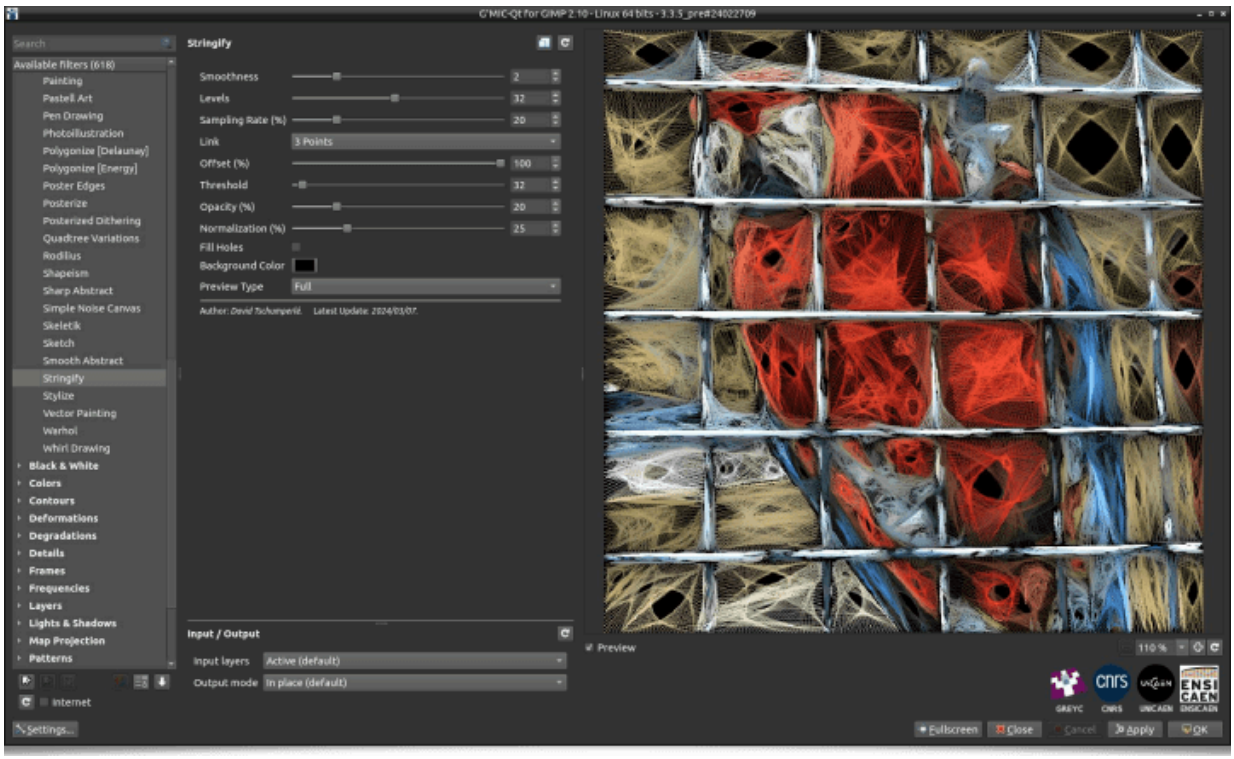


*Fig.3.5.1. The **Artistic / Stringify** filter creates image abstractions from color segments.*

- The **Black & White / Filaments** filter is based on the projection of thousands of particles from one or more edges of the image, with trajectories deformed by the image's contour geometry. The drawing of these thousands of trajectories with semi-transparent colors produces images like these:

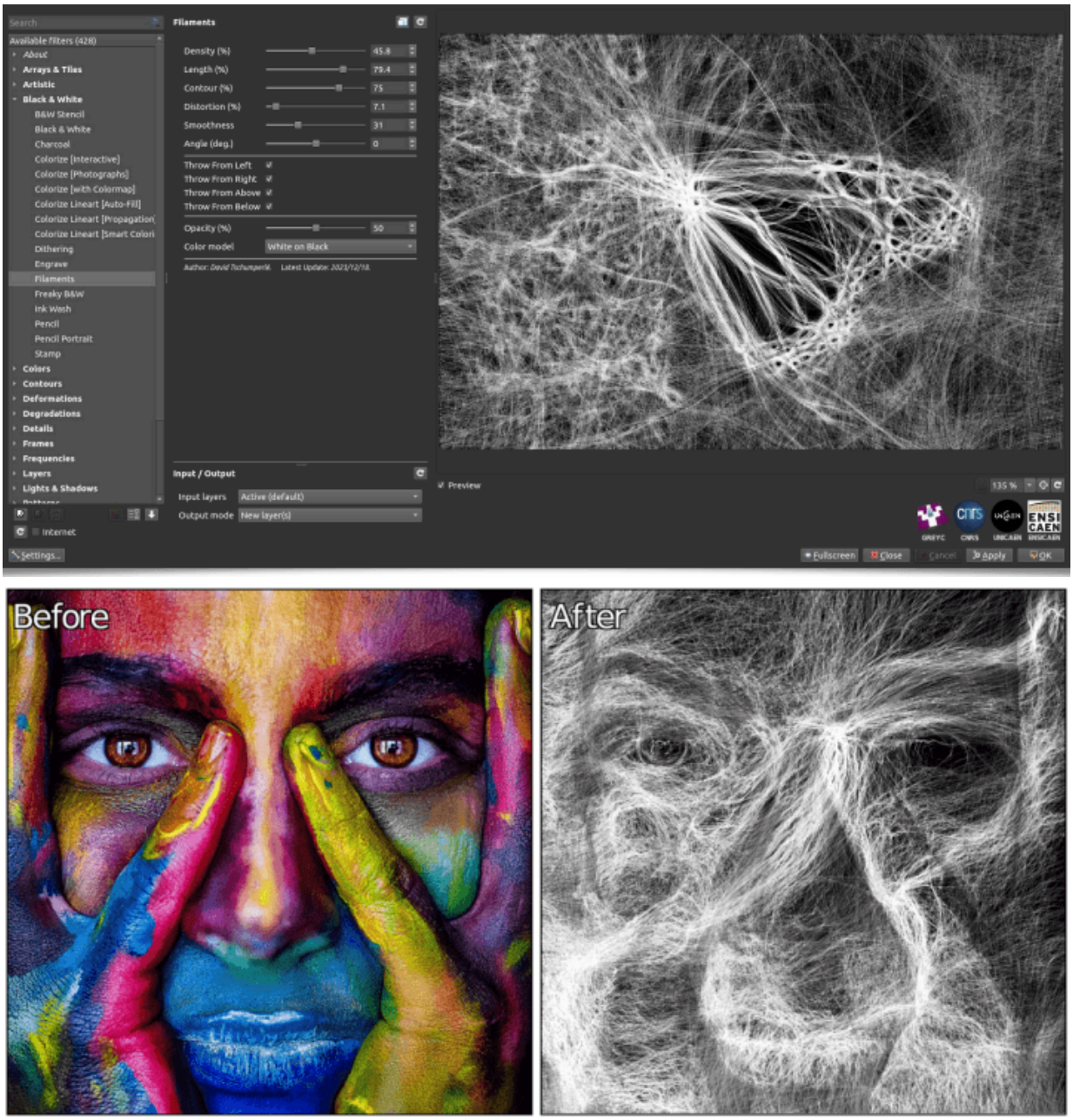


*Fig.3.5.2. The **Black & White / Filaments** filter transforms your images into sets of deformed filaments.*

- The **Arrays & Tiles / Loose Photos** filter simulates an effect of throwing photos haphazardly on a table, so that the content of the photos constitutes a global image, as specified by the filter's settings. Many parameters are adjustable (density, size and ratio of photographs, shading parameters, etc.), giving the user a great deal of freedom over the final rendering.

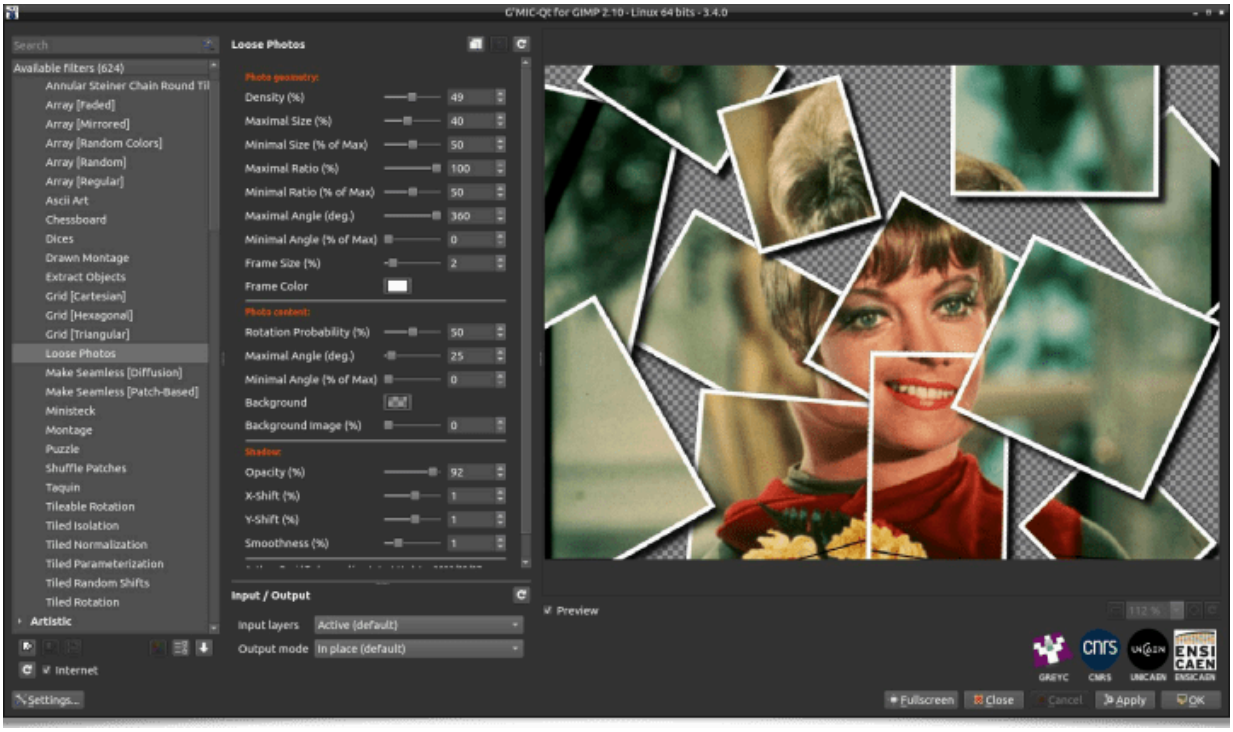


*Fig.3.5.3. Example of rendering by the **Arrays & Tiles / Loose Photos** filter.*

- The **Rendering / Quick Copyright** filter, which already existed in previous versions, has been re-implemented from scratch. It facilitates the insertion of a copyright text (or text signature) on a digital image. It now has many parameters to finely adjust the position of the text, its size, font, and other aspects. Combined with the automation capabilities of the *G'MIC* language, this filter is suited for easily inserting text, in a standardized way, on thousands of photographs.

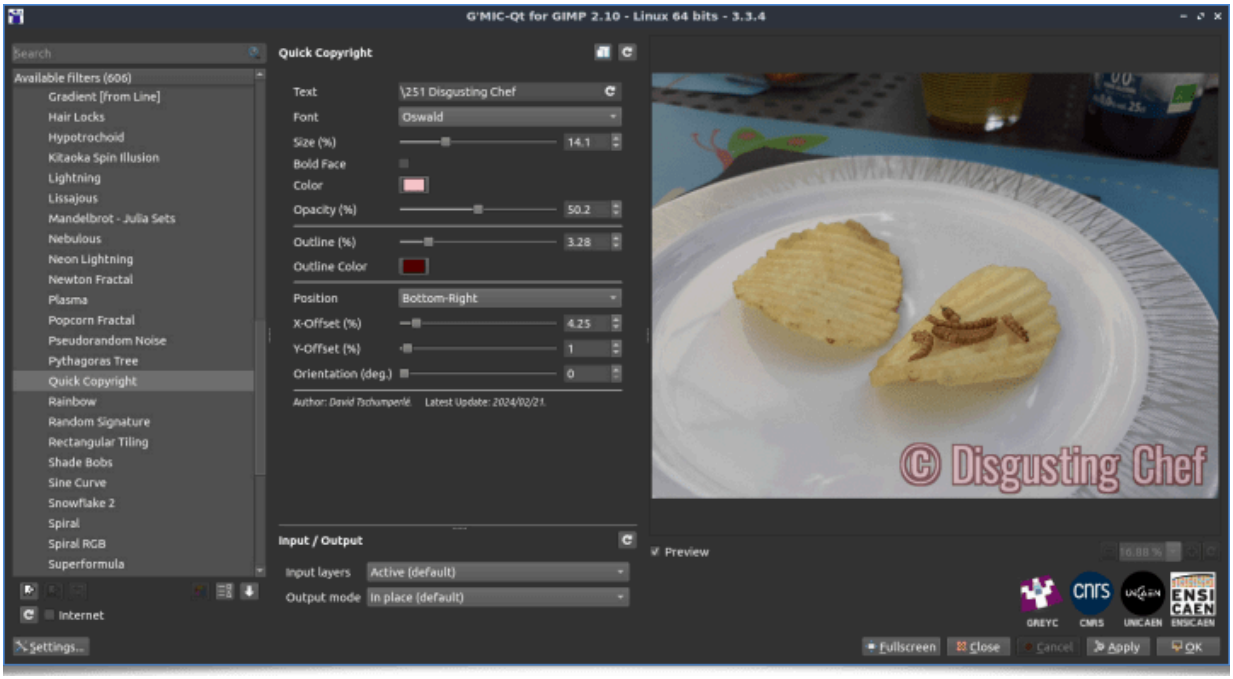


*Fig.3.5.4. The **Rendering / Quick Copyright** filter in action for inserting copyright text on an image.*

- Finally, the **Patterns / Random Rectangles** filter transforms an image into a random partition of colored rectangles, giving an abstract image effect, vaguely in the manner of *Piet Mondrian*.

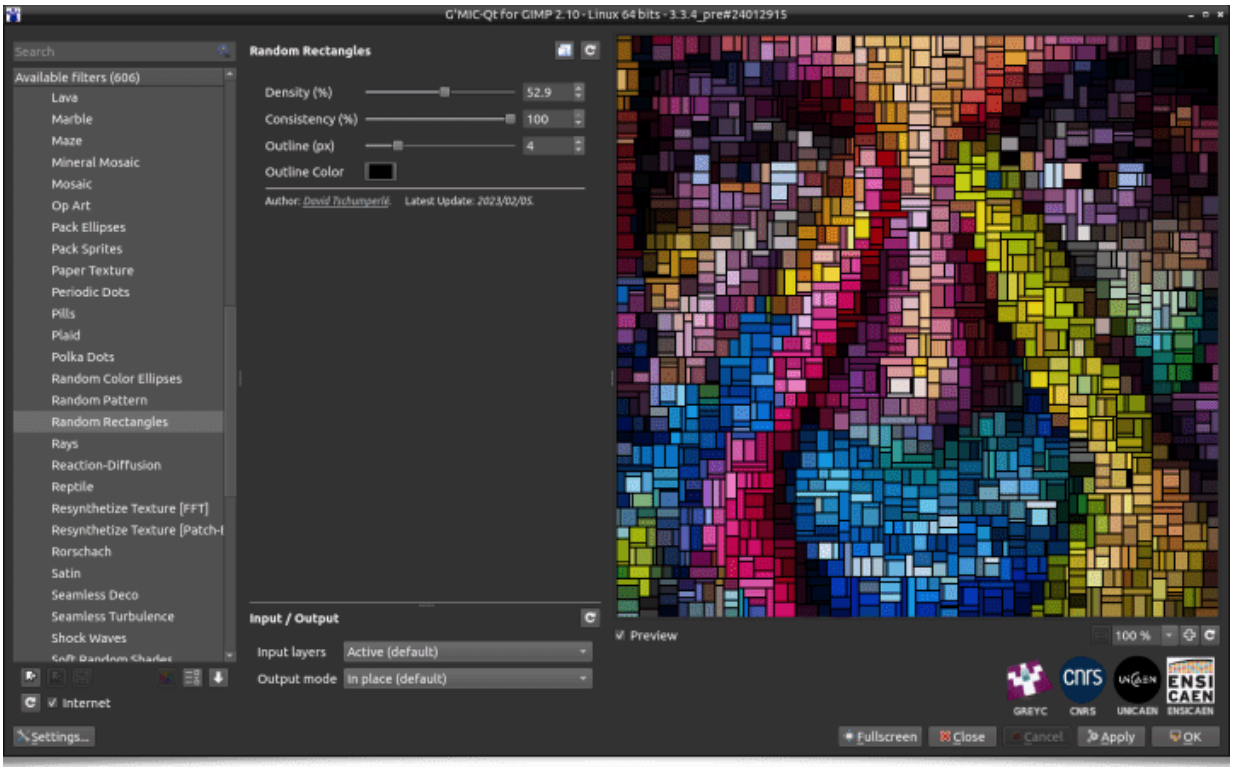
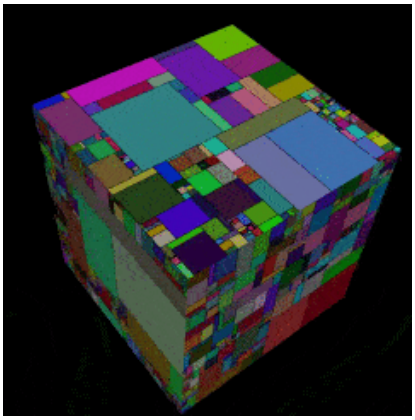


*Fig.3.5.5. The **Patterns / Random Rectangles** filter creates a colorful partition of randomly placed rectangles on the image.*

One can easily imagine using this filter to generate decorative object textures, as demonstrated in this animation below, which starts from a pure noise image:



4. A Software with Varied Uses

As we have shown, *G'MIC* has multiple facets with a wide range of applications. The user community is not limited to digital artists but also includes researchers, programmers, algorithmicians, etc. Below are some other varied uses of the software.

- **Denoising JWST Images:**

In October 2023, we learned from *M.J. McCaughrean*, senior researcher at the *ESA* (*European Space Agency*) (retired in 2024), that *G'MIC* was used by some people within the *ESA* to process images from the *James Webb Space Telescope* (*JWST*), particularly for attenuating the frequency noise appearing on some images acquired by the telescope (using the **Repair / Banding Denoise** filter, among others).

G'MIC was used in conjunction with among other software for creating the cover image of the *Nature* magazine, vol. 622, issue 7981 on October 5, 2023, as confirmed by *M.J. McGaughrean* and indicated in *the associated publication*, of which he is a co-author.

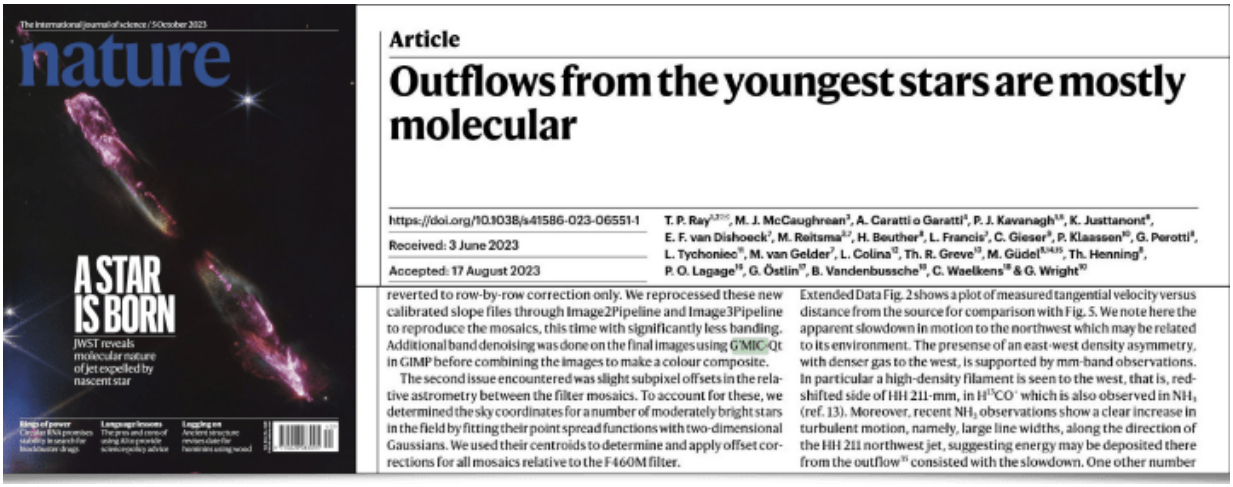


Fig.4.1. Use of G'MIC for denoising images of the Herbig-Haro 211 protostar, acquired by the JWST.

It was a pleasant surprise for us developers to learn this and realize that *G'MIC* was being used in the field of astrophysics.

- **Creative Coding:**

G'MIC proves to be a valuable ally for *creative coding*, due to its ability to easily generate and manipulate images through its standard library of operators. Over the past year, we have enjoyed exploring its possibilities for algorithmic creation of images and animations, a small selection of which is shown here:

Let's start with this little animation of swirling snowflakes (*source code*, 30 lines):

Fig.4.2. Snowflake animation generated by a G'MIC script.

Then let's continue with this amusing variant of the *Rock-Paper-Scissors* game, where each pixel in an image (whose initial random values represent either *0: a rock*, *1: paper*, or *2: scissors*) plays consecutively with its 8 neighbors, keeping the element that won the most. In the second phase, we stylize the sequence of these

different label images, again, with the **Drop Water** filter, resulting in the following animation, which we might call *"Hell's Soup"* (*source code*, 30 lines).

Fig.4.3. Rock-Paper-Scissors game animation, where all the pixels in an image play simultaneously.

Finally, let's dive back into the wonderful world of *Mandelbrot fractals*, starting with this rendering of the Mandelbrot set using the *Orbit Trap* technique, allowing the rendering to be mapped with a color image:

Fig.4.4. Rendering of the Mandelbrot Set using the Orbit Trap method.

Then, with another type of rendering known as *Buddhabrot*, but here considering complex series of the type *z_[n+1]=z_[n]^p + c*, and linearly varying the real exponent *p* between 0 and 6 (rather than keeping *p=2* as for the classic Mandelbrot set), to generate each image of the animation below (source code and details *on this page*):

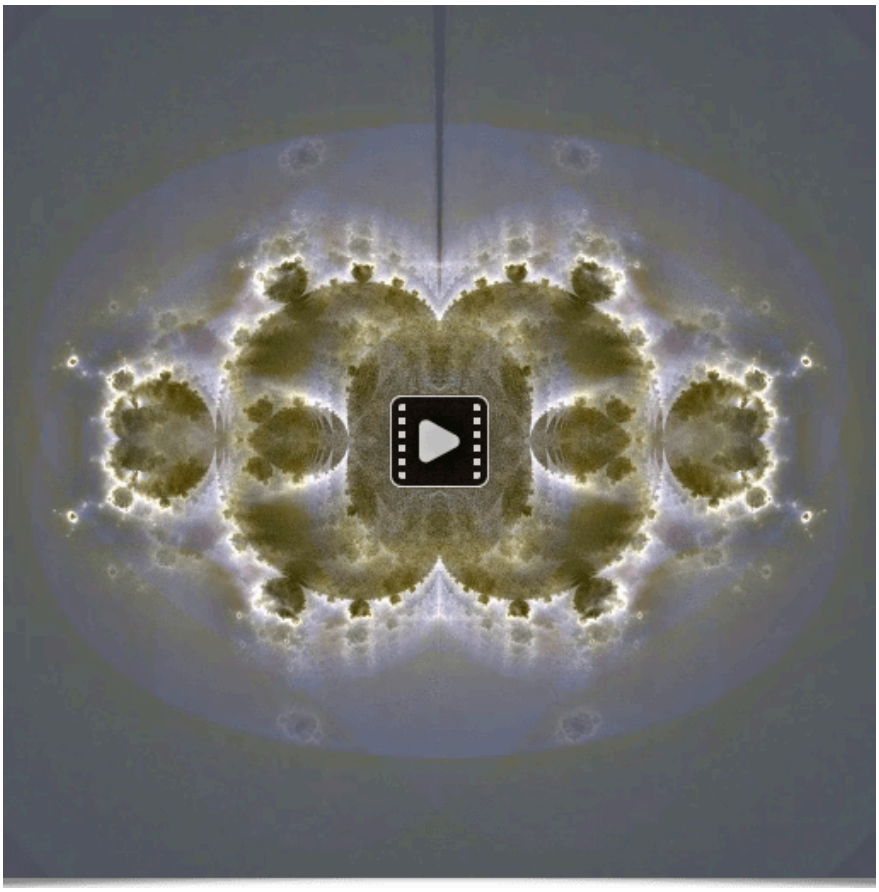


Fig.4.5. Fractal variations around the Buddhabrot.

Note that generating these high-resolution modified *Buddhabrot* images requires significant computation time (a few minutes per image). I find this animation intriguing: it is quite easy to see/hallucinate familiar shapes when you look closely at some frames of the animation, a bit like when you look at clouds in the sky (if *you think you see* in this animation, a bear's head, an old man sitting, a character's silhouette, a dragon's head, ..., then you are not alone in being a bit of a dreamer ☺).

- **Image Conversion for *String Art* **:

String Art is a form of artistic creation characterized by the use of colored threads connected between points (most often nails) to form a pattern or reproduce a photograph (usually portraits). As an experiment, we wrote a *G'MIC* script that tries to transform a user-selected grayscale input image into a series of instructions to follow to connect numbered nails with a monochrome thread, to reproduce the image as faithfully as possible (considering the constraints specific to this mode of creation):

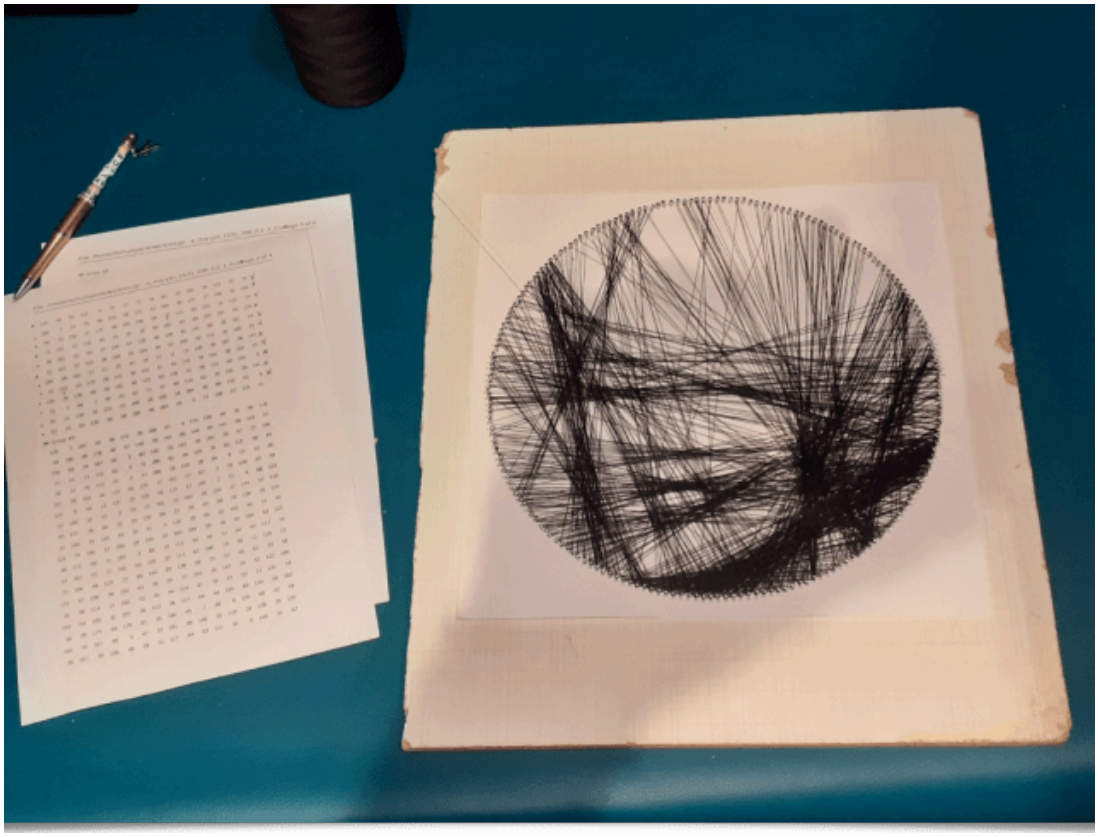


Fig.4.6. More or less successful attempt to use G'MIC for String Art.

Can you recognize who is supposed to be represented in the photo below? An attempt not necessarily crowned with success, which deserves to be explored further (especially by managing threads of different colors), but which nevertheless illustrates the multiple possibilities for experimentation that the *G'MIC* framework allows!

- **Other Related Links**:

To conclude, here are some miscellaneous links that demonstrate the versatility of *G'MIC*:

```
1. **Steganography**: How to hide data in a noise image? The `rand` command can generate [_random
values with density_](https://en.wikipedia.org/wiki/Probability_density_function), meaning their
probability density is a function specified by the user. This has, for example, the application
of hiding 256x256 images in histograms of 16-bit/channel noise images, as [_detailed on this
page_](https://discuss.pixls.us/t/new-function-rand-pdf-random-values-following-a-custom-
distribution).

2. **Artistic Galleries**: [_Ivelieu_](https://deviantart.com/ivelieu/gallery) and [_Gannjondal_]
(https://deviantart.com/gannjondal/gallery) are two artists on the _Deviant Art_ site who
sometimes use _G'MIC_ in their artistic creation process. Take a look at their respective
galleries, it's worth it!
```

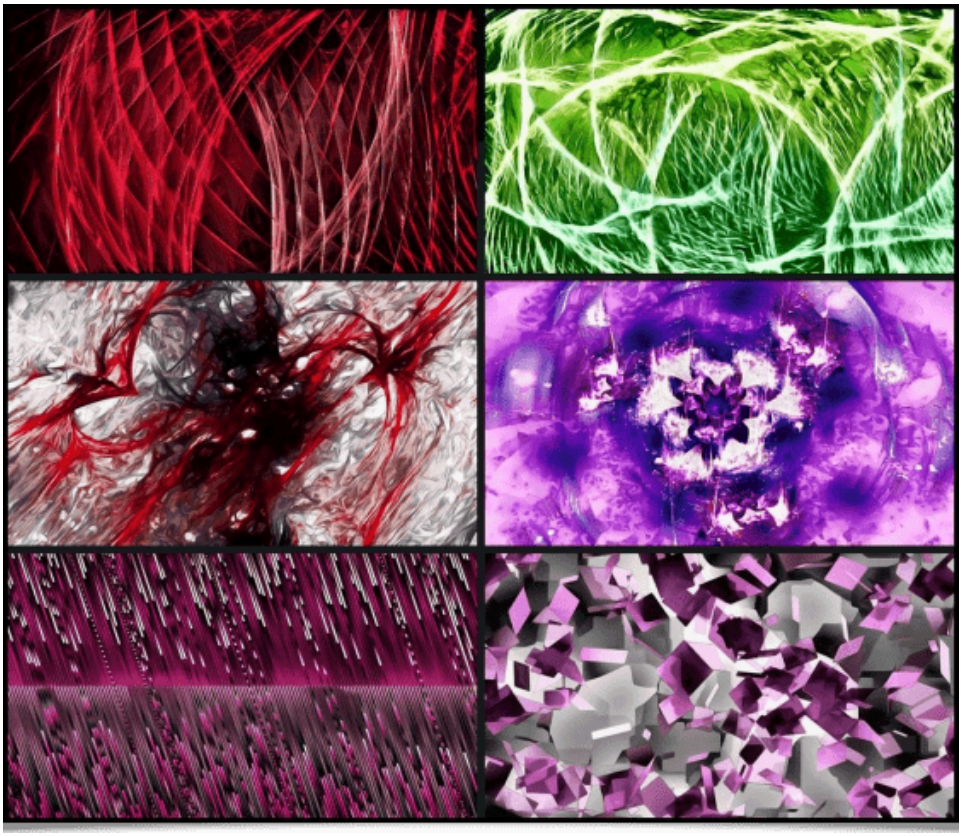


Fig.4.7. Some works by Ivelieu / Deviant Art.

Fig.4.8. Some works by Gannjondal / Deviant Art.

3. We learned from _Gilles Caulier_ that the next version of the photo management program _digiKam_ 8.4.0 [_will include a G'MIC processing tool_](https://www.reddit.com/r/kde/comments/1d67s1d/next_digikam_840_photo_manager_program_will/) within the _Batch Queue Manager_, which will allow _digiKam_ users to integrate _G'MIC_ filters into their post-processing workflows.

4. [_Thiojoe_](https://www.thiojoe.com/) is a YouTuber who produces videos about technology and is a programmer in his spare time. He started developing a free tool based on _G'MIC_ to easily generate animations, a tool that you can [_find here_](https://github.com/ThioJoe/Gmic-Animation-Tools). It only works on _Windows_ for now, but we'll keep an eye on it.

5. Finally, let's mention the _YouTube_ channel of [_JustCallMeInsane_](https://www.youtube.com/@JustCallMeInsane/search?query=g%27mic), a digital illustrator using _Krita_. She recently made a series of videos exploring the different categories of filters in the _G'MIC-Qt_ plugin for _Krita_. We hope in passing that the _Krita_ development team, which manages its own version of the plugin, will be able to update it quickly.

And that's it for the latest news around the *G'MIC* project and its new version *3.4.0*.

5. Conclusions

We could summarize this report by saying that the *G'MIC* project "continues its journey".

G'MIC remains, of course, a modest project, developed and maintained by a small team (of enthusiasts), but with a growing number of users and increasingly varied usage feedback. This framework has the chance to be developed within the *GREYC*, a public research laboratory, supported by the laboratory's supervisors (the *CNRS* "Computer Science" institute, the University of Caen, and *ENSICAEN*), and encouraged by the laboratory's management.

And even if most of the development on this project has probably already been done (more than 15 years of development after all), we are confident that *G'MIC* will continue to evolve; in any case, we will try, as long as it remains useful to image processors of all kinds!