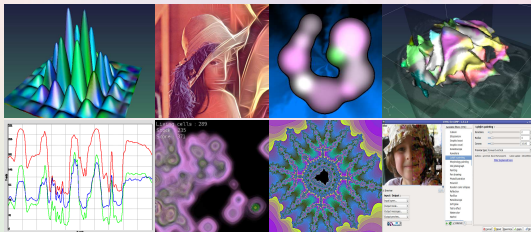




## A Full-Featured Framework for Image Processing



**David Tschumperlé**

Image Team, GREYC / CNRS (UMR 6072), Caen / France

FOSDEM'2013, Brussels/Belgium, February 2013



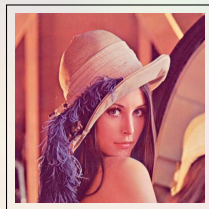
- (Public) Research on Image Processing, at the GREYC lab of the ENSICAEN / CNRS / University of Caen.
- ⇒ We are trying to design (innovative) algorithms to solve problems related to image processing (image denoising, enhancement, segmentation, features detection, ...).

- Frequent collaborations with companies / laboratories having specific image data to process.

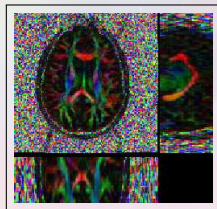


⇒ Various image data coming from very different sources.

- Types of images to process are diverse : 2D, 2D+t, 3D, 3D+t, float-valued, hyperspectral or even matrix-valued pixels/voxels.
- ⇒ Sometimes, we stray far from just 2D color pictures !



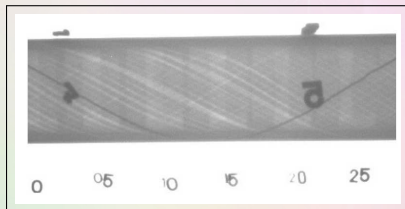
(a)  $I_1 : W \times H \rightarrow [0, 255]^3$



(b)  $I_2 : W \times H \times D \rightarrow [0, 65535]^{32}$



(c)  $I_3 : W \times H \times T \rightarrow [0, 4095]$



(d)  $I_4 : W \times H \times T \rightarrow [0, 4095]$



- **Needs for specific tools** to visualize / explore data, convert image formats, apply classical IP operators (filtering, geometric transformations, frequency analysis, ...) for **very generic** images types, sometimes on several gigabytes of image data.
- **Typical question heard at the lab:** “How may I easily convolve 3d volumetric images with 32 channels, by an anisotropic gaussian kernel ?”



- ⇒ Very few existing open-source tools for these kind of tasks. They tend to be either:
- ▶ Easy to use, but **not generic enough for our data** (ImageMagick, GraphicsMagick, ...).
  - ▶ Or very flexible, but **reserved for savvy programmers** (requires the writing of code, using “complex” external libraries).
- **We did like others:** Since 1999, we have been developing a **C++ library for generic image processing**.



## The Cimg Library

C++ Template Image Processing Toolkit



<http://cimg.sourceforge.net>

CImg is a C++ library which is:

Simplicity

Easy to install and to manipulate.

Genericity

Generic enough to be able to process a wide variety of image types (*2D,3D,3D+t,hyperspectral,float-valued,...*).  
(*template-based*)

Usefulness

Provide *usual algorithms* encountered in the Signal and Image Processing fields.

Extensibility

Extensible by nature.

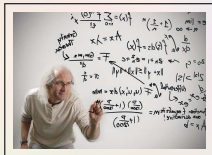
Portability

Portable on several OS and architectures.

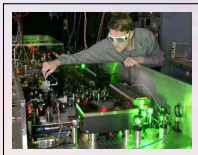
Freedom

Distributed under an *open-source* license.  
(*CeCILL-C*)

- So, problem solved ? No !  $\Rightarrow$  People are also generic ! 😊
- The world of image processing research consists of people with very different profiles:



Mathematicians



Physicists



Geeks



Biologists ...

$\Rightarrow$  Providing a C++ library for Image Processing is **still too restrictive** to reach / help most of these people !



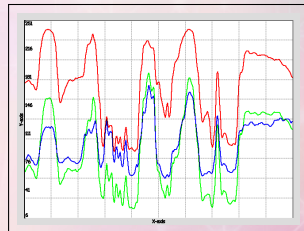
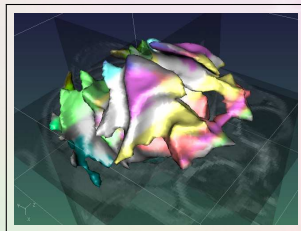
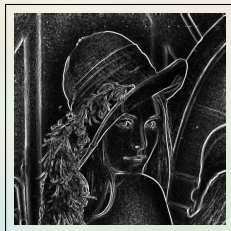
<http://gmic.sourceforge.net>

- **Goals :** G'MIC aims at providing **several user interfaces** to **easily access the image processing features** of the **Clmg Library**.
- Those different interfaces are more or less user-friendly (and powerful) and aim at **different audiences**.
- **Technical means :** G'MIC defines **a whole script language**, specifically designed to build **complex image processing pipelines** in a **concise** way (**G'MIC language**), used as a basis layer in all proposed user interfaces.

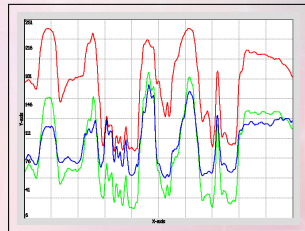
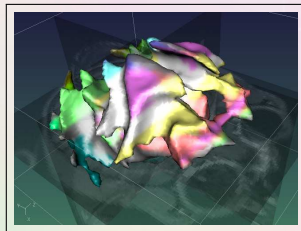
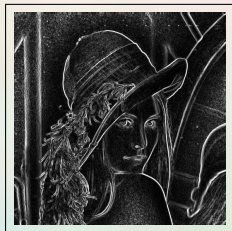
## 1. Definition of a script language designed to build **complex image processing pipelines** (*G'MIC* language).

- ▶ **Full-featured:** More than **750 commands** available (to date) for image visualization, filtering, geometry / color management, features extraction, 3d rendering, matrix computations, graphical plots, ...

→ **Current documentation (.pdf) has more than 300 pages.**



1. Definition of a script language designed to build **complex image processing pipelines** (*G'MIC* language).
    - ▶ **Conciseness:** The G'MIC language has been designed specifically for being **concise**. This is an interpreted language, which can be **extended by custom user-defined functions** (generally short).
- Primary target of use was the command line.





## 2. Provide an **open-source implementation** of the **G'MIC language interpreter** (as a C++ library).

- ▶ **Integrations:** Third-party softwares can easily get all *G'MIC* features (interesting for image retouching or painting softwares, ...).
- ▶ **Free software:** The G'MIC interpreter is distributed under the **CeCILL license** (GPL-compatible).

→ **Very few “external” integration have been done yet:**

- ★ *EKD*, video editing software.
- ★ **Planned:** *Krita* (plug-in), painting software.
- ★ **Planned:** *Delaboratory*, RAW photograph postprocessing application.

## 3. Providing **easy-to-use user interfaces** (also multi-platform), embedding the *G'MIC* language interpreter.

- ▶ **gmic** : Tool to manipulate generic image data from **from the command line (CLI)**. Competitor to the CLI tools of the ImageMagick / GraphicsMagick projects.

```
dtschump@ :"$ gmic ~/work/img/lena.bmp -blur 3 -mirror x
[gmic]-0./ Start G'MIC parser.
[gmic]-0./ Input custom commands file '/home/dtschump/work/src/resources.gmic' (added 14 commands, total 1121).
[gmic]-0./ Input custom commands file '/home/dtschump/work/src/gmic/src/gmic_def.gmic' (added 1107 commands, total 2228).
[gmic]-0./ Set dynamic 3d rendering mode to flat-shaded.
[gmic]-0./ Input file '/home/dtschump/work/img/lena.bmp' at position [0] (1 image 512x512x1x3).
[gmic]-1./ Blur image [0], with standard deviation 3 and neumann boundary.
[gmic]-1./ Mirror image [0] along the 'x'-axis.
[gmic]-1./ Display image [0] = 'lena.bmp*'.
lena.bmp* (512x512x1x3) ; this = 0xbf9852e4, size = 1/16 [3072 Kb], data = (Cimg<float*>)0xa027a44..0xa027a5b.
  [0] : this = 0xa027a44, size = (512,512,1,3) [3072 Kb], data = (float*)0xb73ba008..0xb76ba007 (non-shared) = [ 203,2
86 207,053 210,367 212,452 212,914 211,713 209 205,179 ... 65,5009 63,9167 62,7955 61,9959 61,279 60,5754 59,9074 59,3
564 ], min = 9,43401, max = 250,19, mean = 128,229, std = 55,711, coords_min = (511,440,0,1), coords_max = (68,57,0,0)
+
[gmic]-1./ End G'MIC parser.
dtschump@ :"$ █
```

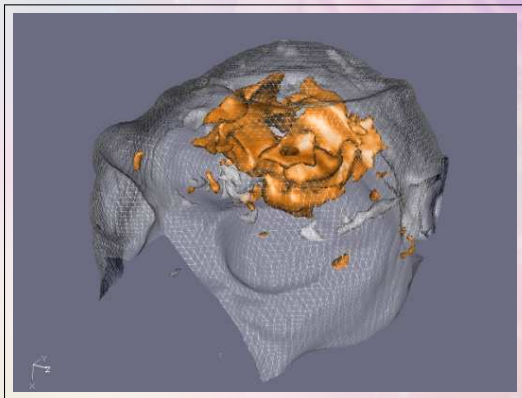
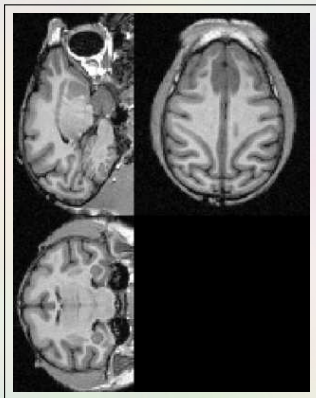
# Example of use for 'gmic'

```
gmic lena.bmp -blur 3 -sharpen 1000 -noise 30 -+  
" 'cos(x/3)*30' "
```



# Example of use for 'gmic'

```
gmic reference.inr -flood 23,53,30,50,1,1,1000 -flood[-2]  
0,0,0,30,1,1,1000 -blur 1 -isosurface3d 900 -opacity3d[-2] 0.2  
-color3d[-1] 255,128,0 +3d
```

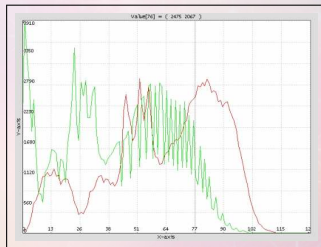


# Example of use for 'gmic'

```
gmic milla.bmp -f '255*(i/255)^1.7' -histogram 128,0,255 -a c -plot
```

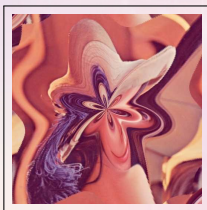
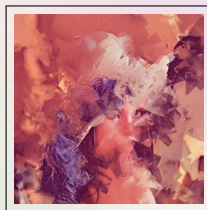
is the G'MIC equivalent to this C++ code (using CImg):

```
#include "CImg.h"
using namespace cimg_library;
int main(int argc, char **argv) {
    const CImg<>
    img("milla.bmp"),
    hist = img.get_histogram(128,0,255),
    img2 = img.get_fill("255*((i/255)^1.7)", true),
    hist2 = img2.get_histogram(128,0,255);
    (hist,hist2).get_append('c').display_graph("Histograms");
    return 0;
}
```



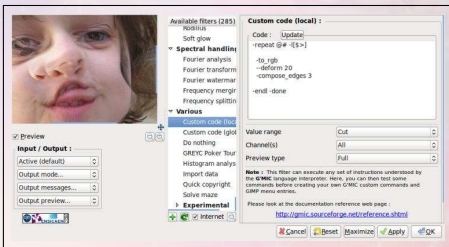
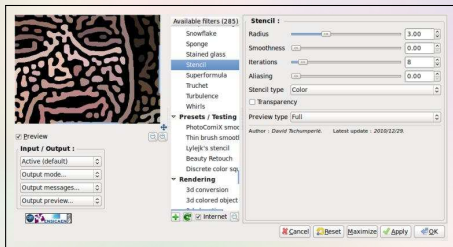
- A *G'MIC*-written pipeline can be added as a new *G'MIC* command.
- Writing pipelines **also** allows creation of nice artistic filters !

```
gmic lena.jpg -pencilbw 0.3 -o gmic_lena1.jpg; gmic lena.jpg  
-cubism 160 -o gmic_lena3.jpg  
gmic lena.jpg -flower 10 -o gmic_lena4.jpg; gmic lena.jpg  
-stencibw 30 -o gmic_lena2.jpg
```



## 3. Providing **easy-to-use user interfaces** (also multi-platform), embedding the *G'MIC* language interpreter.

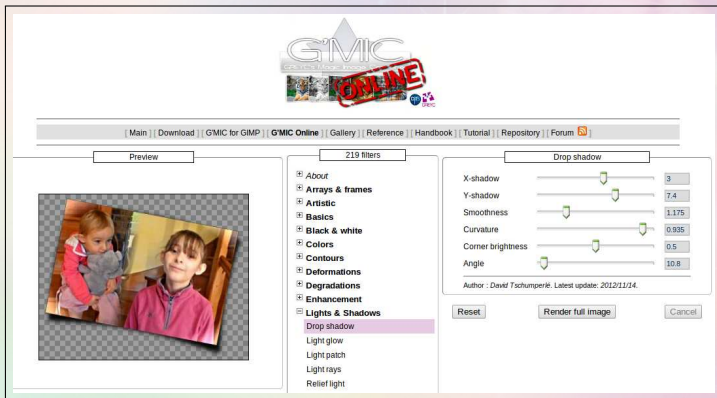
- ▶ **gmic\_gimp** : Plug-in for GIMP provides hundreds of image filters on 2D RGB or RGBA images.





## 3. Providing **easy-to-use user interfaces** (also multi-plateform), embedding the *G'MIC* language interpreter.

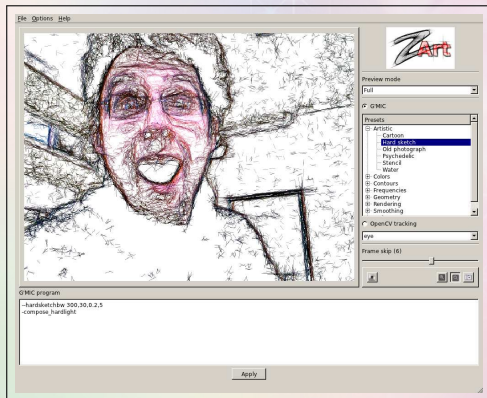
- ▶ **G'MIC Online** : Web service for **manipulating images online** (similar to the GIMP plug-in, but running on a web browser).  
<https://gmicol.greyc.fr>

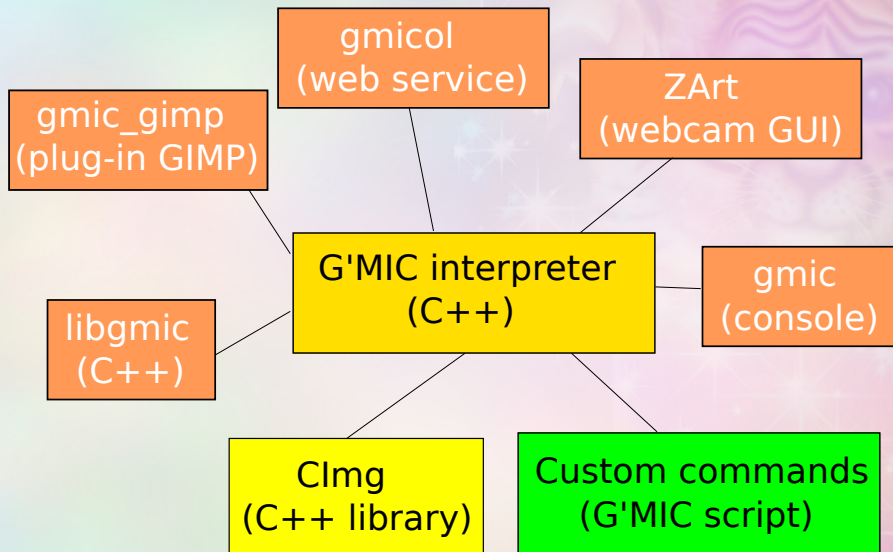


The screenshot displays the G'MIC Online web interface. At the top, there is a navigation bar with links: [ Main ] [ Download ] [ G'MIC for GIMP ] [ **G'MIC Online** ] [ Gallery ] [ Reference ] [ Handbook ] [ Tutorial ] [ Repository ] [ Forum ]. Below the navigation bar, the interface is divided into three main sections:

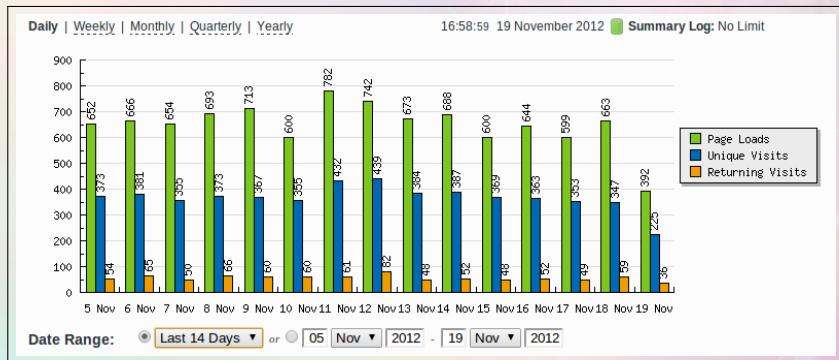
- Preview:** Shows a photo of two children with a drop shadow effect applied. The image is tilted and has a semi-transparent shadow cast behind it.
- 219 filters:** A list of filter categories including About, Arrays & frames, Artistic, Basics, Black & white, Colors, Contours, Deformations, Degradations, Enhancement, and Lights & Shadows. The 'Drop shadow' filter is currently selected and highlighted.
- Drop shadow:** A configuration panel for the selected filter. It includes sliders and input fields for:
  - X-shadow: 3
  - Y-shadow: 7.4
  - Smoothness: 1.175
  - Curvature: 0.935
  - Corner brightness: 0.5
  - Angle: 10.8At the bottom of this panel, there are buttons for 'Reset', 'Render full image', and 'Cancel'. A small text note at the bottom of the panel reads: 'Author : David Tschumperlé. Latest update: 2012/11/14.'

3. Providing **easy-to-use user interfaces** (also multi-plateform), embedding the *G'MIC* language interpreter.
  - ▶ **ZArt** : A QT-based interface for manipulating images acquired from the webcam (used as a demonstration platform).





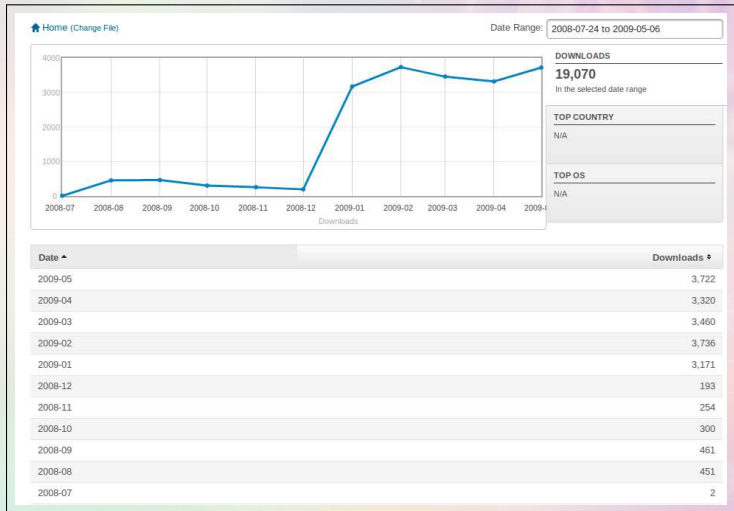
- Today, the *G'MIC* project has:
  - ▶ A little less than 100,000 lignes de code (mainly in C++ and G'MIC languages).
  - ▶ 250-350 downloads / day (+ than 700.000 since July 2008).
  - ▶ 350-400 unique visitors / day on the project web page.
- ⇒ Very satisfactory statistics regarding the focused audience...
- ⇒ Digital artists have also invested in the project !



# Major evolution step

- Sources/binaries of the GIMP plug-in, have been made available in **January 2009**.

⇒ Made a big difference in the number of downloads a day.



- Some features that have gained G'MIC attention:
1. One of the few open-source software to propose an efficient image denoising algorithm:



- Some features that have gained G'MIC attention:
1. One of the few open-source software to propose an efficient image denoising algorithm:





- Some features that have gained G'MIC attention:
2. One of the few open-source software to propose an **image inpainting** algorithm:



- Some features that have gained G'MIC attention:
2. One of the few open-source software to propose an **image inpainting** algorithm:



- Some features that have gained G'MIC attention:
2. One of the few open-source software to propose an **image inpainting** algorithm:



- Some features that have gained G'MIC attention:
2. One of the few open-source software to propose an **image inpainting** algorithm:



- **Some features that have gained G'MIC attention:**
- 2. One of the few open-source software to propose an **image inpainting** algorithm, here for B&W image recolorization using color patches:



(Courtesy of Akros/GimpChat)

- Some features that have gained G'MIC attention:
3. 'Fractalius'-like effect (39\$ plug-in for Photoshop), reproduced with G'MIC 'Rodilius' (→ 0\$, 10 lines of G'MIC code !):



Redfield Fractalius



G'MIC Rodilius



```
1. rodilius :
2. -v - -repeat @# -l[$>] -split_opacity -rv
3. -if {!$6} -negative[-1] -endif
4. --f[-1] 0 -nm[-1] @{-2,n}
5. -repeat {round($4)}
6.   angle=({$5+$>*180/round($4)})
7.   --blur_linear[-2] $1%,{$1*$2/100}%, $angle,1 -b[-1] 0.7 -sharpen[-1] $3 -max[-2,-1]
8. -done -rm[-2]
9. -if {!$6} -negative[-1] -endif
10. -rv -a c -endl -done -v +
```



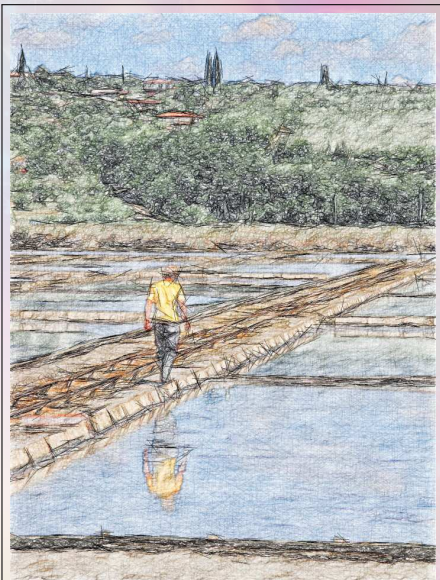


- Some features that have gained G'MIC attention:
4. Original 'Sketch' effect, available in G'MIC.
- Publication in the **IEEE International Conference on Image Processing**, in 2011.



(Courtesy of Tom Keil)

# Sketch results



(Courtesy of Tom Keil)





(Courtesy of Tom Keil)

- G'MIC defines a lot of ways to play with images of any types.
- G'MIC is a generic framework with several different interfaces.

⇒ Go try it ! ☺

## Thanks for listening!

Any questions are welcome...

