G’MIC : 2.2, v’là les filtres !

Posté par **David Tschumperlé** (**site web personnel**) le 16/02/18 à 20:41. Édité par **5 contributeurs**. Modéré par **ZeroHeure**. **Licence CC By-SA.**

Étiquettes : **g'mic** , **traitement_d'images** **+ Étiqueter**

L’équipe IMAGE du laboratoire **GREYC** (UMR **CNRS** 6072), situé à Caen, est ravie de vous annoncer la sortie d’une nouvelle version (numérotée **2.2**) de ***G’MIC***, son cadriciel libre, générique et extensible pour le **traitement des images** [W].

Comme **à notre habitude**, nous profitons de cette occasion pour faire un point sur les dernières fonctionnalités notables ajoutées depuis la version majeure précédente (**2.0**), sortie qui avait fait l’objet d’une **dépêche** sur *LinuxFr.org*, en juin 2017.

Sommaire

N. D. A. : Cliquez sur les images de la dépêche pour en visualiser des versions à meilleure résolution.

1. Contexte et évolutions récentes du projet

G’MIC est un projet libre développé depuis août 2008 (distribué sous licence **CeCILL**), dans l’équipe **IMAGE** du laboratoire **GREYC**, laboratoire de recherche public français localisé à Caen et chapeauté par trois tutelles : le **CNRS**, l’**Université de Caen**, et l’**ENSICAEN**. Cette équipe est composée de chercheurs et d’enseignants-chercheurs spécialisés dans les domaines de l’algorithmique et des mathématiques du traitement d’images.

*Fig. 1.1 : Logo du projet G'MIC, cadriciel libre pour le traitement d'images, et sa mignonne petite mascotte « Gmicky » (élaborée par **David Revoy**).*

G'MIC est multi-plate-forme (GNU/Linux, macOS, Windows…) et fournit un ensemble d'interfaces utilisateur variées pour la manipulation de données images *génériques*, à savoir des images ou des séquences d'images hyperspectrales 2D ou 3D à valeurs flottantes (ce qui inclut de fait les images couleur « classiques »). Plus de **950 fonctions** différentes de traitement d'images sont déjà disponibles dans ce cadriciel, nombre extensible à l'infini puisque les utilisateurs ont la possibilité de développer et d'ajouter leurs propres fonctionnalités via l'utilisation du langage de script intégré, spécifiquement dédié à la réalisation de *pipelines* de traitement d'images.

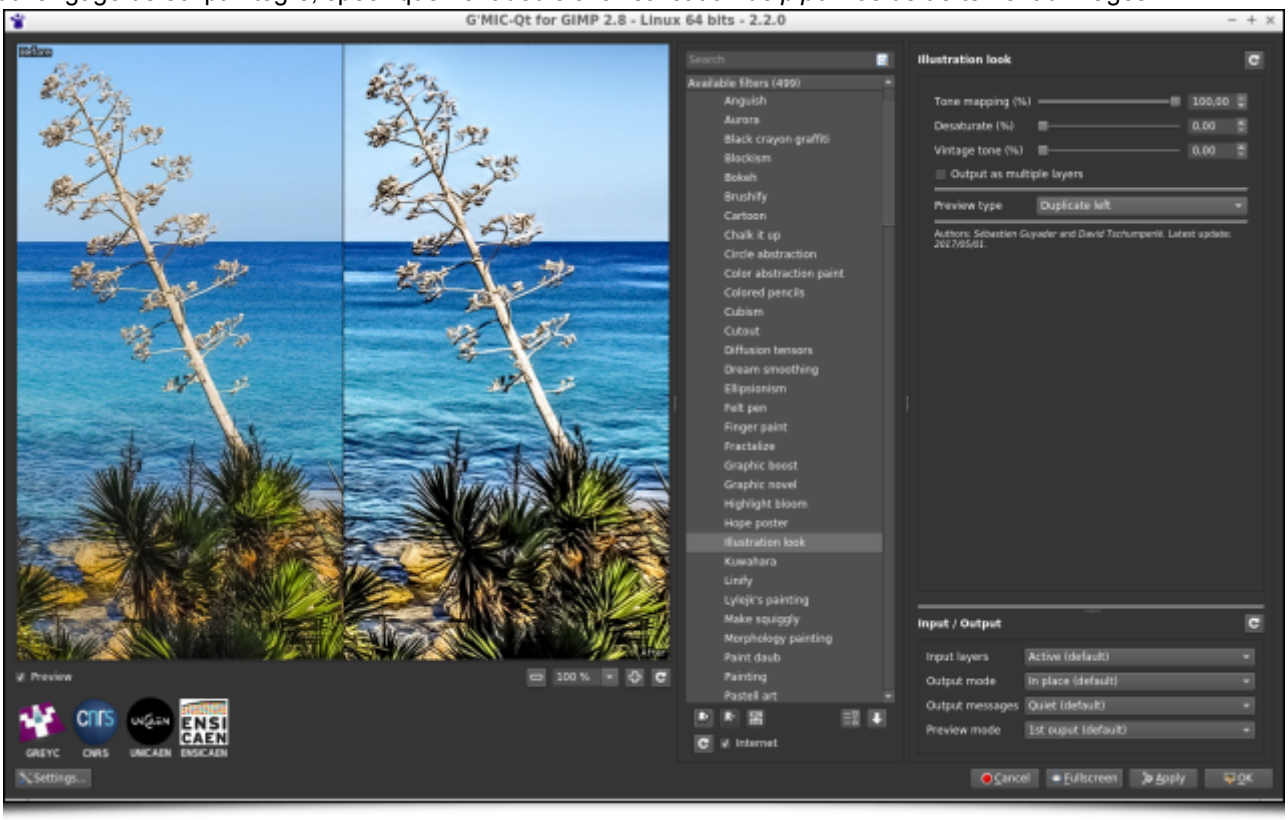


Fig.1.2 : Le greffon G'MIC-Qt *pour GIMP, aujourd'hui l'interface utilisateur de G'MIC la plus populaire.*

Deux événements d'importance dans la vie du projet ont eu lieu, depuis la sortie de la dernière version majeure :

1.1. Portage du greffon G'MIC-Qt vers Krita

Lors de la sortie de la version 2.0, nous étions heureux d'annoncer une refonte complète (en *Qt*[W]) du code du greffon *G'MIC* pour **GIMP**, de façon à disposer d'un greffon *G'MIC-Qt* théoriquement « universel », c'est-à-dire déclinable pour fonctionner sur d'autres logiciels hôtes que GIMP. Eh bien, une étape supplémentaire a été franchie, puisque ce greffon a été effectivement étendu pour s'intégrer dans le logiciel libre de peinture numérique **Krita**.

Ceci a été rendu possible grâce au travail de développement de **Boudewijn Rempt** (mainteneur principal de Krita) et de **Sébastien Fourey** (développeur du greffon). Le greffon *G'MIC-Qt* est aujourd'hui disponible pour les versions 3.3+ de Krita et, même s'il n'implémente pas encore toutes les fonctionnalités d'entrées-sorties de son équivalent pour GIMP, les retours d'utilisation que nous avons eu sont plutôt positifs.

Ce nouveau portage remplace *de facto* l'ancien greffon *G'MIC* pour Krita qui n'était plus maintenu depuis quelque temps. La bonne nouvelle pour les utilisateurs et les développeurs de Krita, c'est qu'ils disposent maintenant d'un greffon dont le code est commun avec celui fonctionnant sous GIMP, et dont nous allons pouvoir assurer la plus grosse partie de maintenance et de mise à jour.

À noter que ce portage a nécessité principalement l'écriture d'un fichier source `host_krita.cpp` (en *C++*) permettant la communication entre le logiciel hôte et le greffon, et on peut raisonnablement penser qu'un effort similaire permettrait à d'autres logiciels de disposer eux aussi de leur propre version du greffon *G'MIC*, et des quelques 500 filtres de traitement d'images livrés avec ! Avis aux développeurs…

Fig. 1.3 : Aperçu du greffon G'MIC-Qt tournant sous Krita.

1.2. Une licence CeCILL-C, plus permissive

Seconde nouvelle importante, la licence d'utilisation ***CeCILL-C*** (licence dans l'esprit de la ***LGPL***[W]) est maintenant proposée pour certains composants du cadriciel *G'MIC*. Cette licence est plus permissive que la licence ***CeCILL*** (qui, elle, est compatible ***GPL***[W]) qui s'appliquait jusque ici, et plus adaptée à la distribution de bibliothèques logicielles. Cette extension de licence (double licence) concerne justement le cœur de calcul de *G'MIC* via sa bibliothèque *C++ libgmic*. Cette nouvelle licence autorise l'intégration des fonctionnalités de la *libgmic* (donc, de tous ses algorithmes de traitement d'images) dans des logiciels non licenciés en GPL/CeCILL (incluant les logiciels à sources fermées).
Le code source du greffon *G'MIC-Qt*, quant à lui, reste distribué sous licence unique CeCILL.

2. Collaboration féconde avec David Revoy

Si vous avez suivi nos **épisodes précédents**, vous avez peut-être remarqué que nous citons très souvent l'artiste illustrateur **David Revoy** pour ses contributions multiples à *G'MIC* : dessin de la mascotte, idées de filtres, tutoriels écrits ou vidéo, tests en tout genre, etc. Plus généralement, David est un contributeur important au monde de l'art numérique libre, autant avec la bande dessinée ***Pepper & Carrot***, qu'il élabore (distribuée sous licence libre *CC-BY*), qu'avec ses suggestions et rapports de bogues continuels pour les logiciels libres qu'il utilise.
Il paraît donc tout naturel de lui consacrer une section spéciale dans cette dépêche, relatant les différentes idées, contributions et expérimentations qu'il a apportées à *G'MIC* tout récemment. Un grand merci à lui pour sa disponibilité, le partage de ses idées, et tant qu'on y est, l'ensemble de son œuvre.

2.1. Perfectionnement du filtre de colorisation de dessins au trait

Évoquons tout d'abord les progrès apportés au filtre ***Black & White / Colorize lineart [smart-coloring]***, filtre qui était apparu lors de la sortie de la version 2.0 de *G'MIC*. C'est un filtre d'aide à la colorisation en aplats de dessins au trait (également dénommés *line art*), qui avait été élaboré en collaboration avec David. Le principe de ce filtre était de générer automatiquement un calque de colorisation aléatoire pour un dessin donné, à partir de l'analyse fine des contours présents dans le calque d'entrée contenant le dessin au trait.
En suivant les suggestions de David, nous avons pu ajouter un nouveau mode de colorisation : le mode *Auto-*

clean. Ce dernier permet de « nettoyer » automatiquement un calque de coloriage (réalisé grossièrement) fourni par l'utilisateur en plus du *line art*, en utilisant la même analyse géométrique des contours que pour les modes de colorisation précédents. L'utilisation de ce nouveau mode est illustré ci-dessous, où un dessin donné (à gauche) a été colorisé de manière approximative par l'utilisateur. À partir des deux calques *line art + colorisation grossière*, le mode *Autoclean* du filtre de colorisation va générer l'image (de droite), où les couleurs ne débordent plus des contours (même des contours « virtuels » non fermés !). Le résultat n'est pas toujours parfait, mais permet néanmoins de réduire le temps passé au processus fastidieux de colorisation.

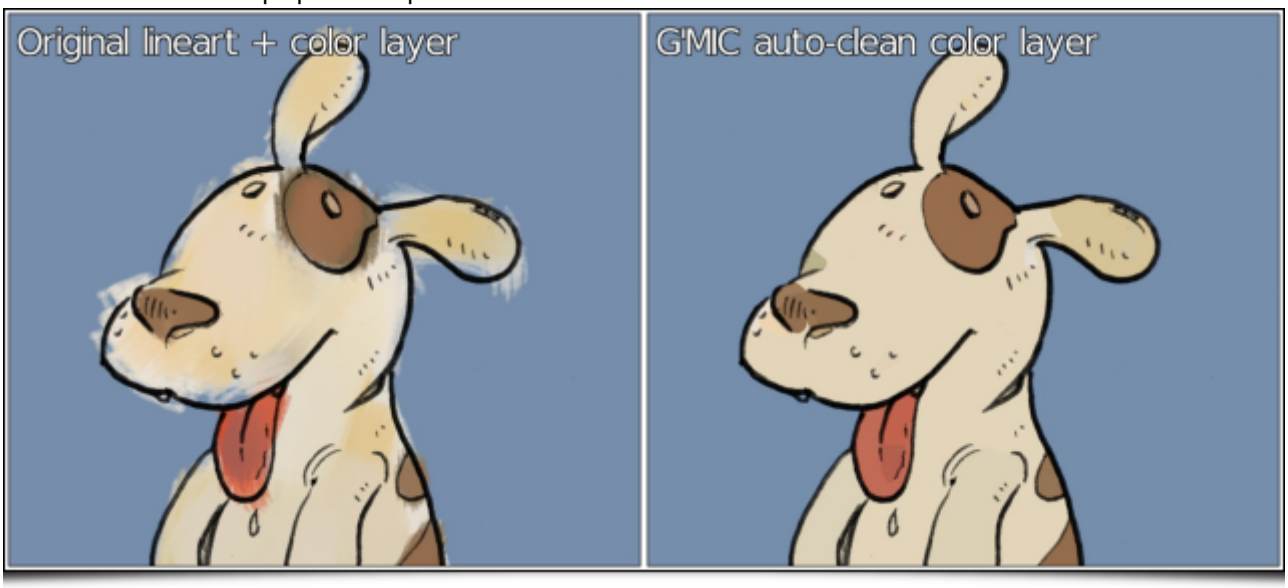


Fig. 2.1 : Le nouveau mode Autoclean *du filtre de colorisation de dessin au trait permet de nettoyer automatiquement un calque d'aplats de couleurs peint approximativement.*

À noter que ce même filtre se dote également d'un nouveau module de détection de hachures, qui permet de ne pas générer trop de petites régions en utilisant l'ancien mode de colorisation aléatoire, lorsque le dessin au trait comporte un nombre important de hachures (résultat de droite sur la figure ci-dessous). Assurément du temps gagné pour l'assignation des couleurs finales par l'artiste !

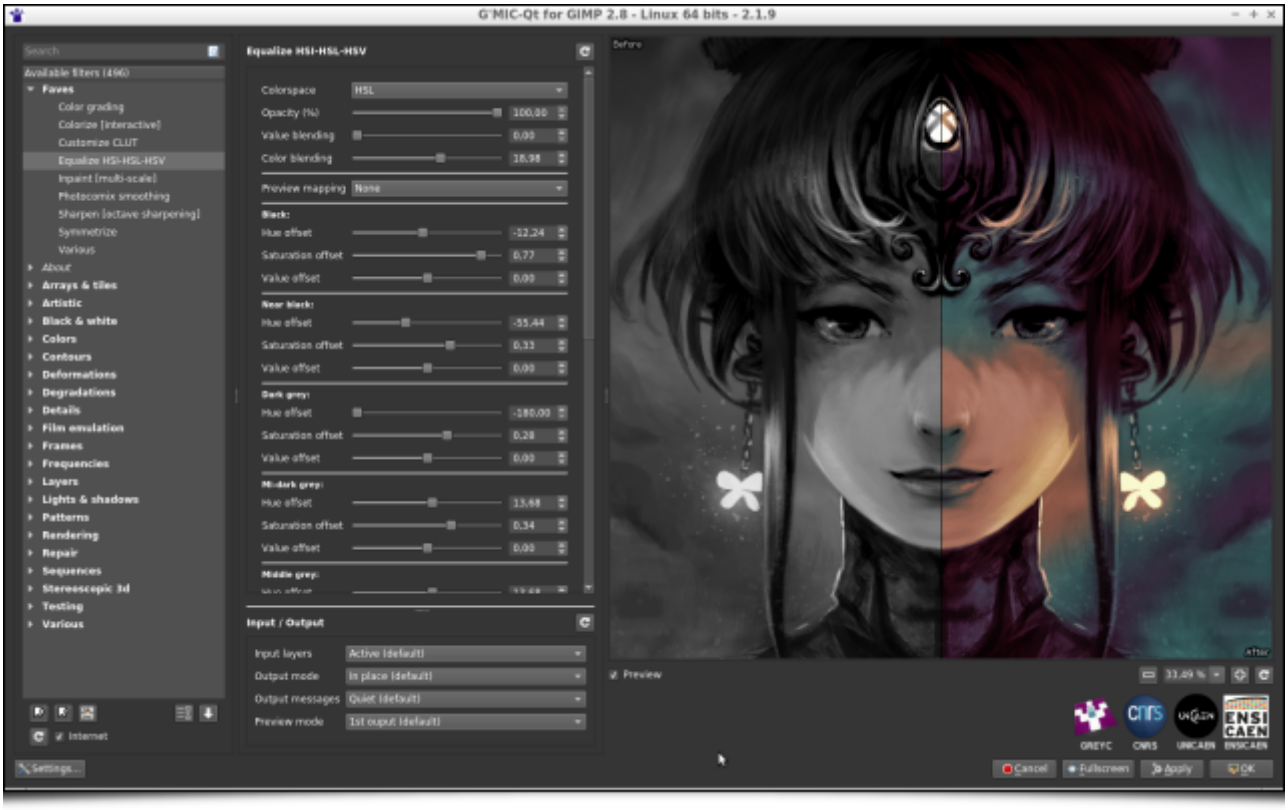
Fig. 2.2 : Le nouveau module de détection de hachures limite le nombre de petites zones de couleurs inutiles générées par le mode de colorisation aléatoire automatique.

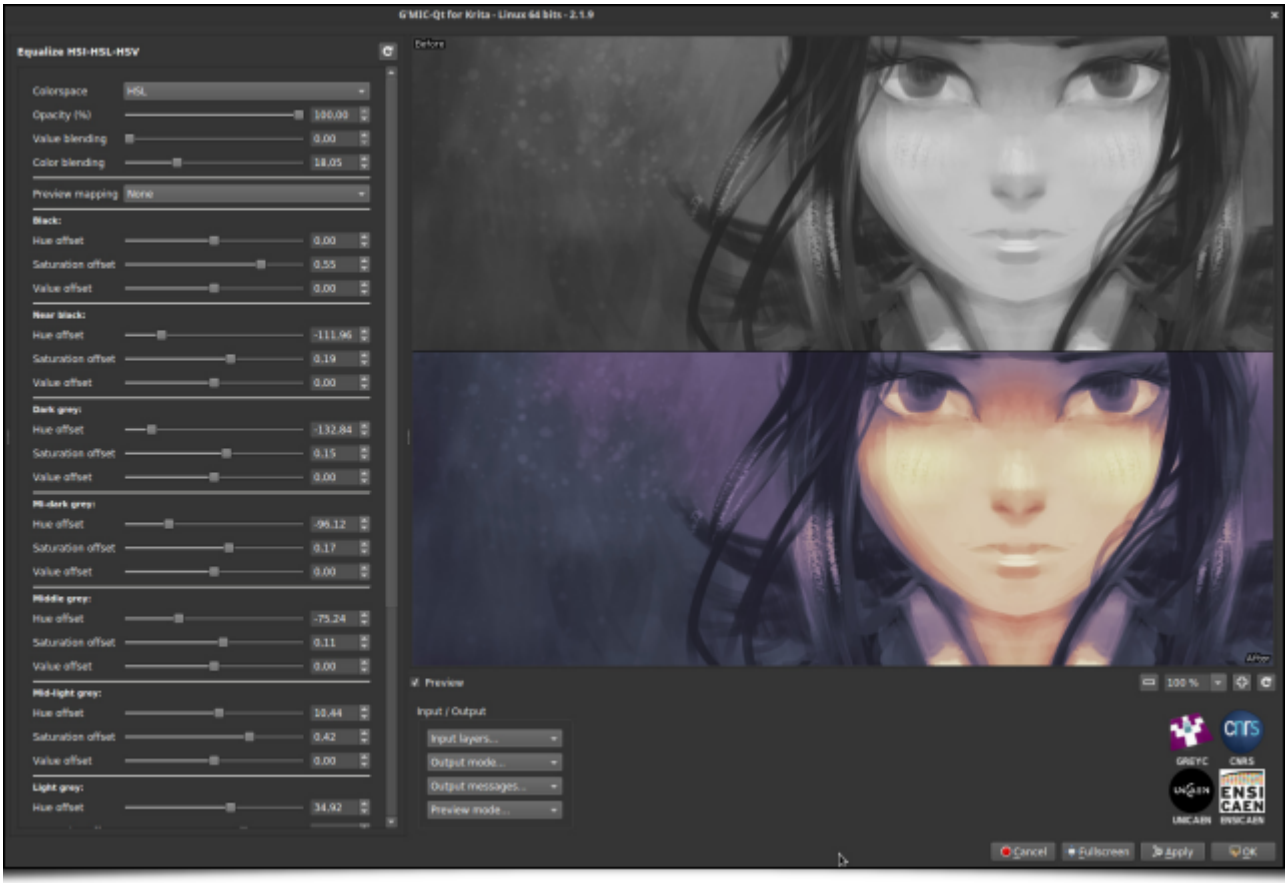
2.2. Égaliseur dans les espaces colorimétriques HSI, HSL et HSV

Plus récemment, David nous a suggéré l'idée d'un filtre permettant de faire varier séparément les teintes et les saturations des couleurs possédant certains niveaux de luminosité. L'idée sous-jacente est de donner à l'artiste la possibilité de dessiner ou peindre numériquement en utilisant seulement des niveaux de gris, puis de coloriser après coup son chef d'œuvre simplement en assignant des couleurs particulières aux différentes valeurs de gris de l'image. La version couleur obtenue possède une palette certes limitée, mais dont l'ambiance colorimétrique globale est déjà en place. L'artiste n'a plus qu'à retoucher localement les couleurs plutôt que d'avoir à coloriser l'ensemble du dessin à la main.

La figure ci-dessous illustre l'utilisation de ce nouveau filtre égaliseur *Colors / Equalize HSI/HSL/HSV* via le greffon *G'MIC* : chaque catégorie de valeurs peut être ajustée finement, avec pour résultats des colorisations préliminaires de dessins en noir et blanc assez bluffants ! On reconnaît dans tous ces exemples la patte artistique de

David, qui a lui-même effectué les tests de ce filtre.





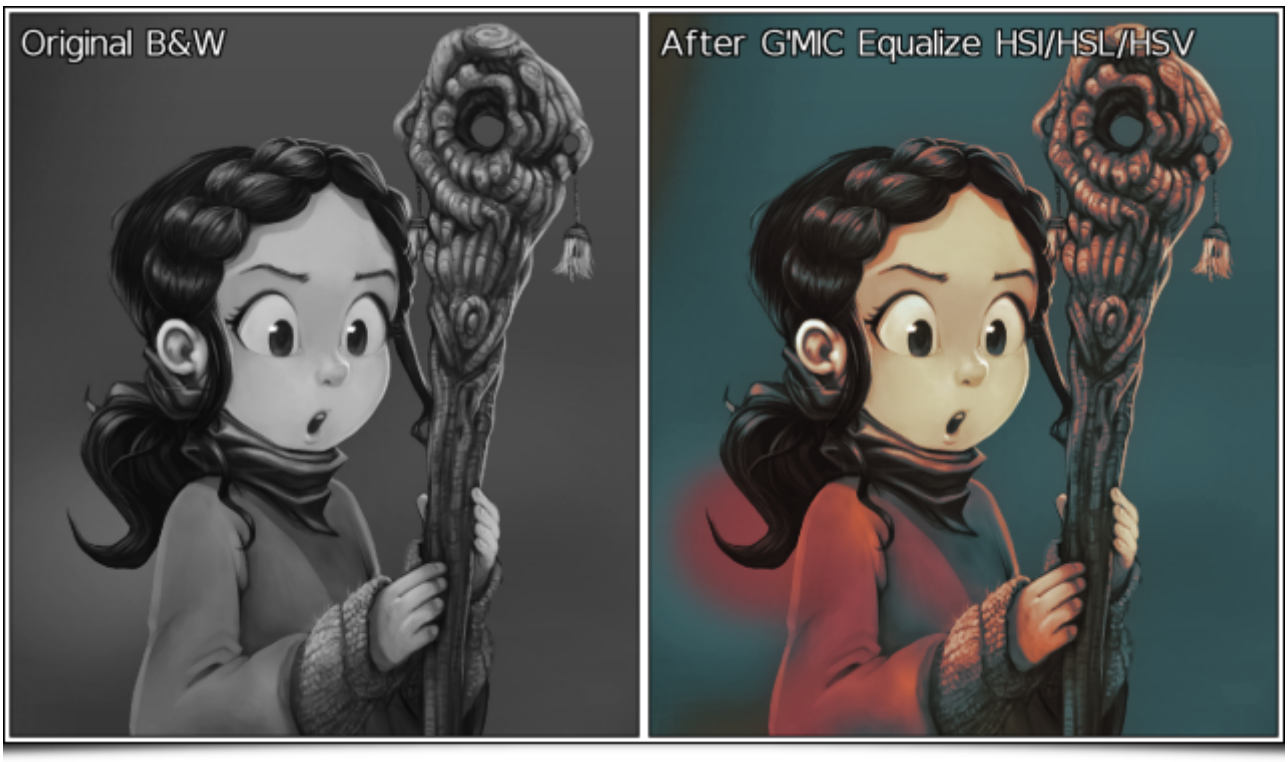


Fig. 2.3 : Le filtre Equalize HSI/HSL/HSV *permet d'assigner des teintes quelconques à des pixels de différentes valeurs de gris.*

Notons que dans le cas d'une image d'entrée en niveaux de gris, l'effet est équivalent à appliquer un gradient de couleurs aux différentes valeurs de l'image, ce qui pouvait déjà se faire assez facilement dans GIMP. Mais l'intérêt ici est de s'assurer que la luminosité des pixels reste inchangée lors de la transformation couleur appliquée, ce qui n'est pas une propriété évidente à préserver lorsqu'on effectue ce type de traitement manuellement via la définition d'un gradient couleurs.

Et ce qui est sympa avec ce filtre, c'est qu'il peut aussi s'appliquer aux photographies déjà en couleurs. On peut ainsi modifier la teinte et la saturation de couleurs possédant une certaine luminosité, avec un effet pouvant être parfois surprenant, comme celui obtenu sur la photographie de paysage ci-dessous.

Fig. 2.4 : Le filtre Equalize HSI/HSL/HSV *appliqué sur une photographie couleur permet de changer son ambiance colorimétrique, ici de manière assez extrême.*

2.3. Déformations anguleuses

Une autre demande de David a concerné l'élaboration d'un filtre de déformation locale aléatoire d'image, ayant la particularité de générer des déformations anguleuses. D'un point de vue algorithmique, ça semblait relativement simple à réaliser.
Notons qu'une fois l'implémentation effectuée (en style concis : **12 lignes !**) et intégrée dans les mises à jour officielles, David a juste eu à appuyer sur le bouton *Mettre les filtres à jour* de son greffon *G'MIC* pour Krita et le nouvel effet *Deformations/Crease* est apparu dans sa liste de filtres, avec la possibilité pour lui de le tester immédiatement. C'est aussi ça le côté pratique de *G'MIC* !

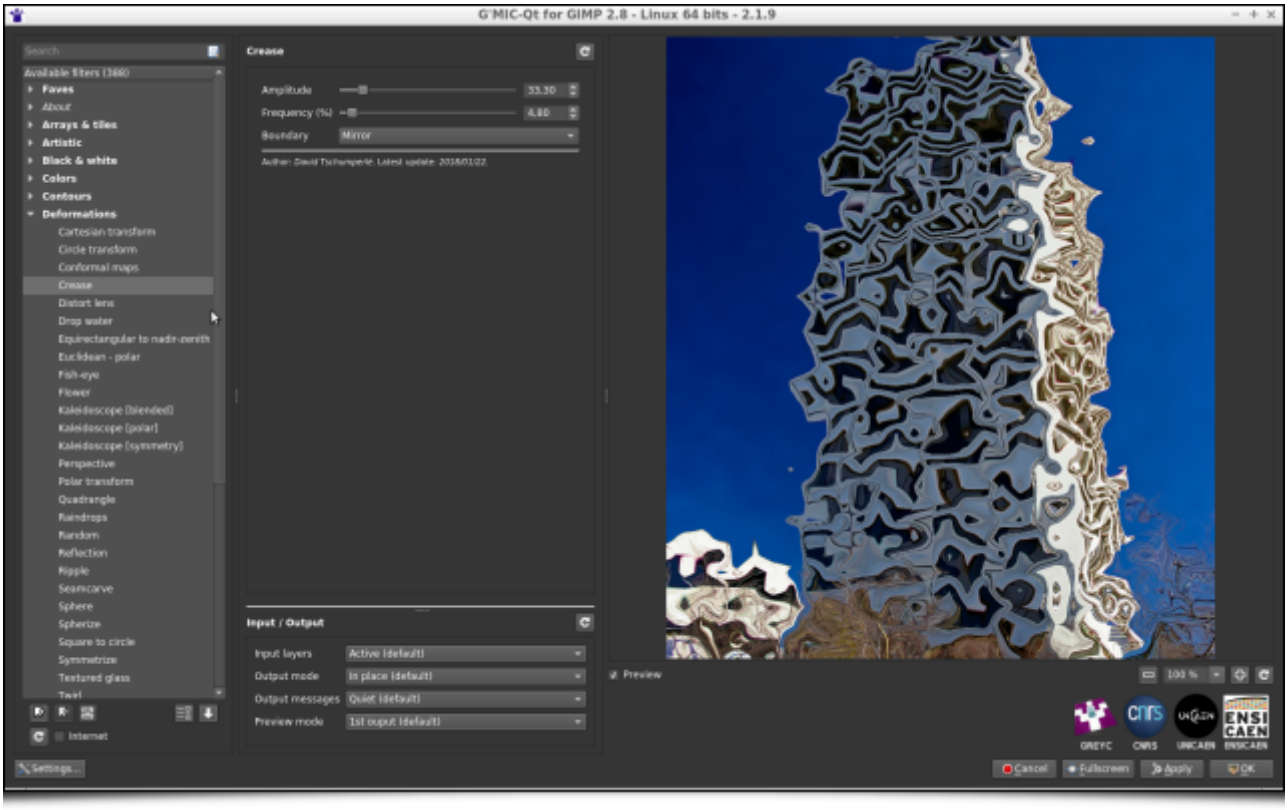
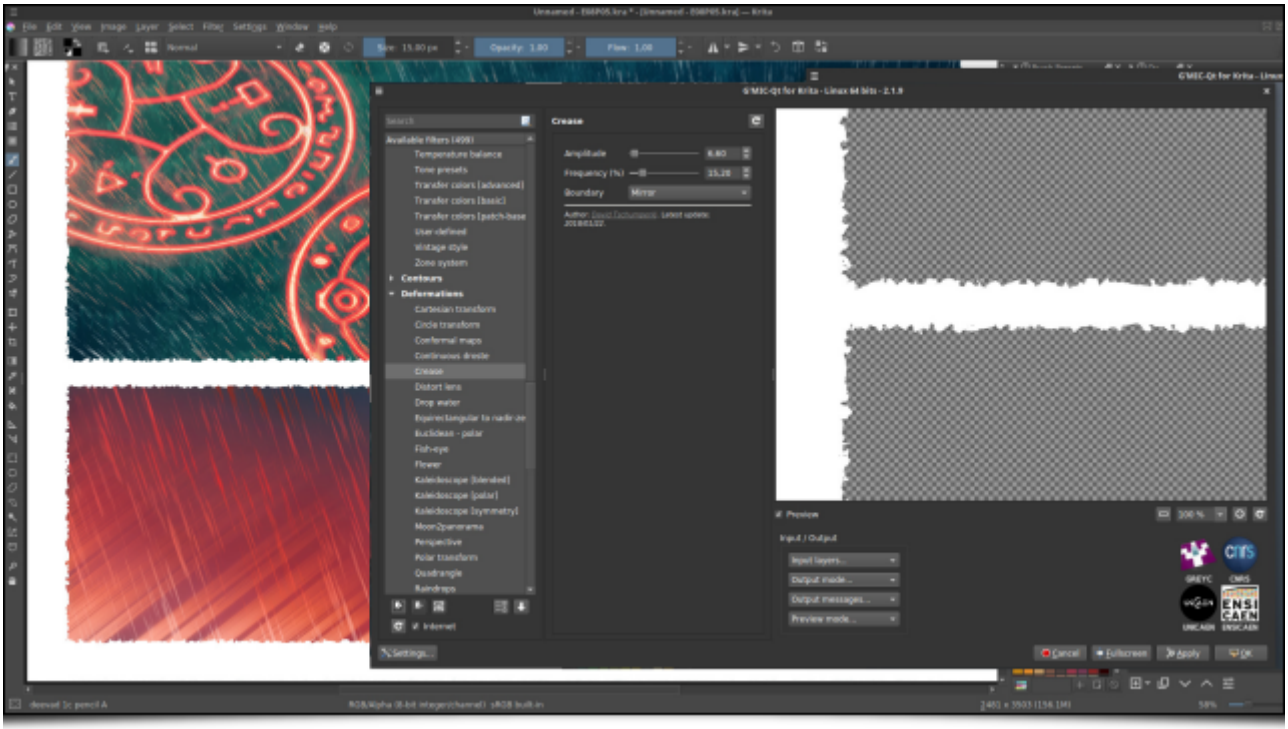


Fig. 2.5 : Nouvel effet Crease *pour des déformations locales anguleuses.*

En revanche, on ne savait pas franchement à quoi ce truc pouvait bien servir en pratique. Mais ce qui est bien quand on coopère avec David, c'est que, lui, il sait très bien ce qu'il va en faire ! Et sur ses expérimentations à lui, ça a tout de suite plus d'allure. En témoigne les deux utilisations qu'il en a fait ci-dessous, d'une part pour donner un aspect crénelé aux bords de certaines de ses cases de BD, et d'autre part pour améliorer le rendu d'un « rayon *alien* de la mort ».



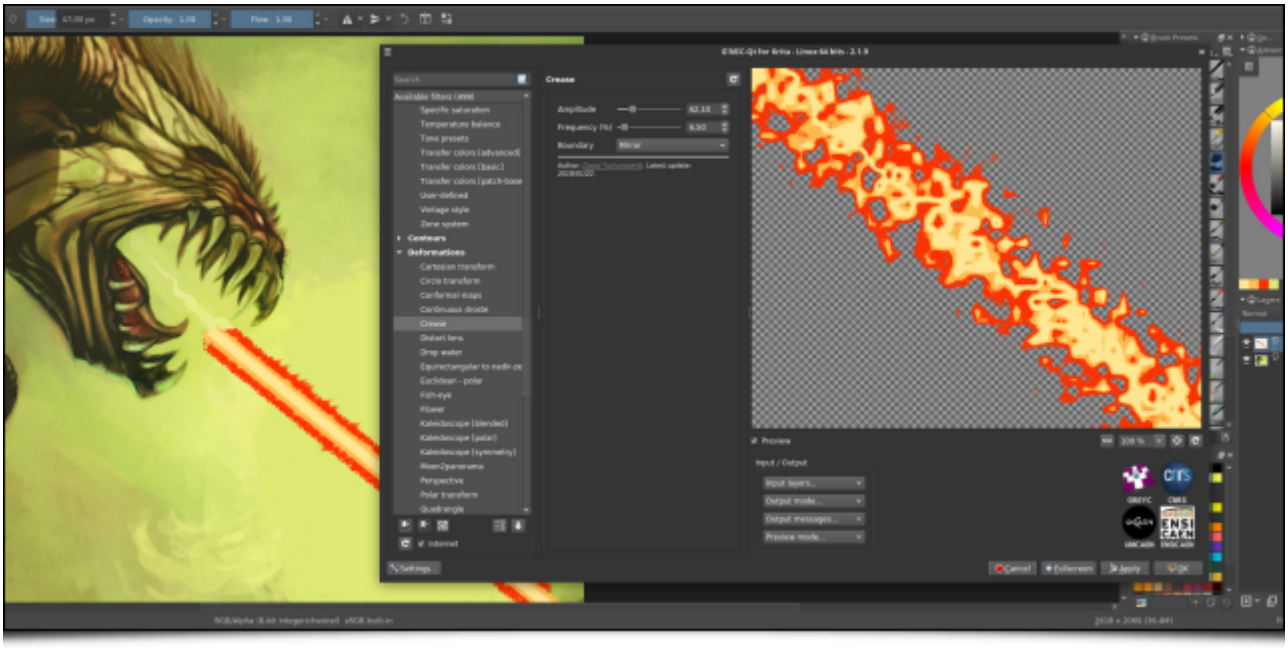


Fig. 2.6 : Utilisation du filtre Crease *pour deux cas concrets de création artistique.*

3. Toujours plus de filtres…

David Revoy n'est *a priori* pas le seul utilisateur de *G'MIC*, nous observons en effet entre 600 et 900 téléchargements quotidiens depuis le site principal du projet. Et il arrive, bien sûr, que d'autres utilisateurs emballés inspirent de nouveaux effets à ajouter, en particulier lors des discussions qui ont lieu sur notre **forum** principal, aimablement mis à disposition par la communauté ***PIXLS.US***.

3.1. Faire ressortir les détails sans créer de « halos »

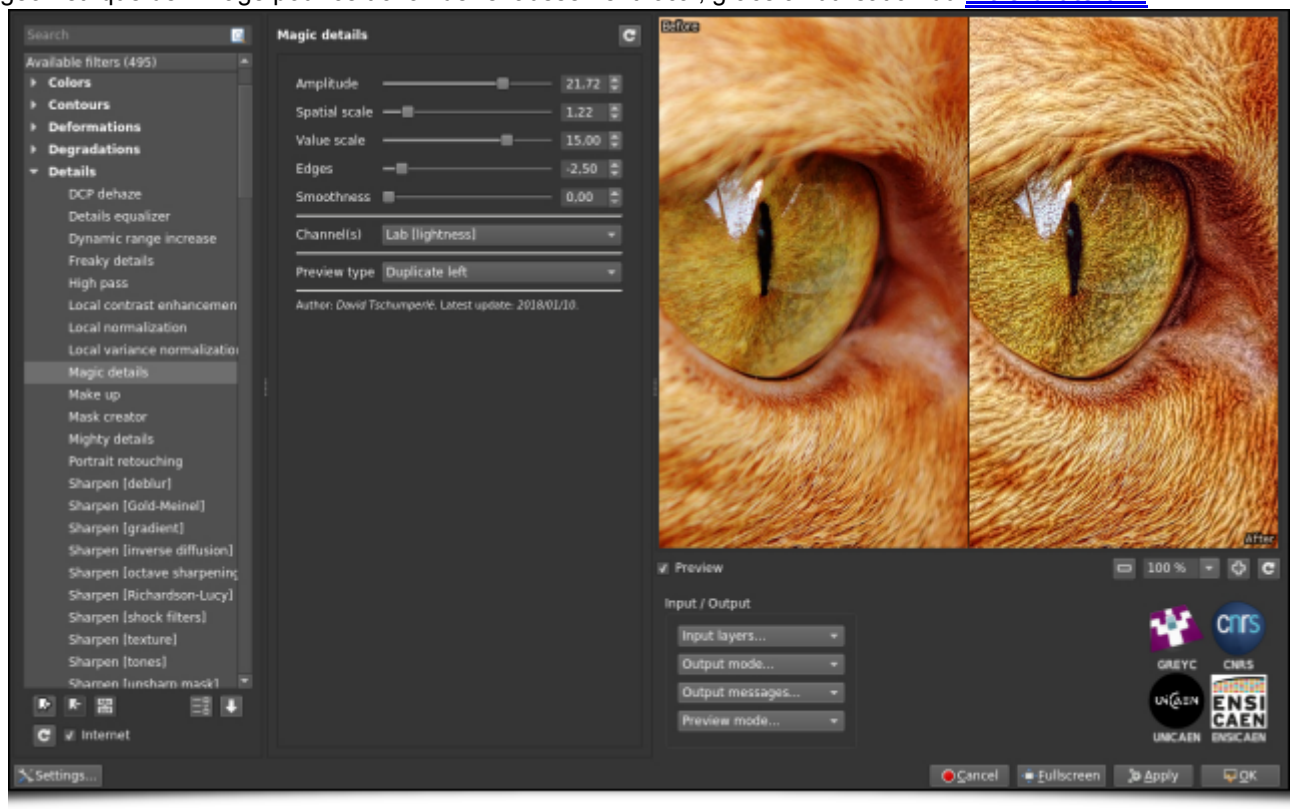
Beaucoup de photographes vous le diront : il n'est pas toujours évident de rehausser les détails dans des photographies numériques sans créer par ailleurs de vilains **artéfacts** [W] qu'il faut souvent remasquer « à la main » après coup.

Il faut savoir que les algorithmes de rehaussement de contraste classiques se basent le plus souvent sur l'augmentation de la variance locale des intensités lumineuses des pixels, voire sur l'égalisation de leurs histogrammes locaux. Malheureusement, ces opérations se font généralement en considérant des voisinages à taille et géométrie fixées, où chaque pixel d'un voisinage est toujours considéré avec le même poids dans les calculs statistiques liés à ces algorithmes.

C'est plus simple et plus rapide, mais d'un point de vue qualitatif ce n'est pas une excellente idée : on se retrouve couramment avec des effets de « halos » autour des contours qui étaient déjà très contrastés dans l'image. Ce phénomène classique est illustré par le cas d'école ci-dessous avec l'application du filtre *Unsharp mask* (présent par défaut dans GIMP) sur une partie d'image de paysage montagneux, et qui génère un « halo » indésirable à la frontière entre montagne et ciel (particulièrement visible sur les images en pleine résolution).

Fig. 3.1 : Des effets de « halos » indésirables apparaissent souvent avec les filtres usuels de rehaussement de contraste.

Tout l'enjeu des algorithmes de rehaussement de détails est donc d'être capable d'analyser la géométrie locale des structures dans les images de manière plus fine, pour prendre en compte des « poids » locaux plus adaptés pour chaque pixel des voisinages considérés. Pour simplifier, on veut rendre **anisotropes** [W] ces méthodes de rehaussement, en les orientant par les contours présents dans les images.
C'est en suivant cette voie que deux nouveaux filtres *G'MIC* ont fait leur apparition récemment, à savoir *Details/Magic details* et *Details/Equalize local histograms*, qui essayent de mieux prendre en compte le contenu géométrique de l'image pour ce travail de rehaussement local, grâce à l'utilisation du **filtre bilatéral** [W].



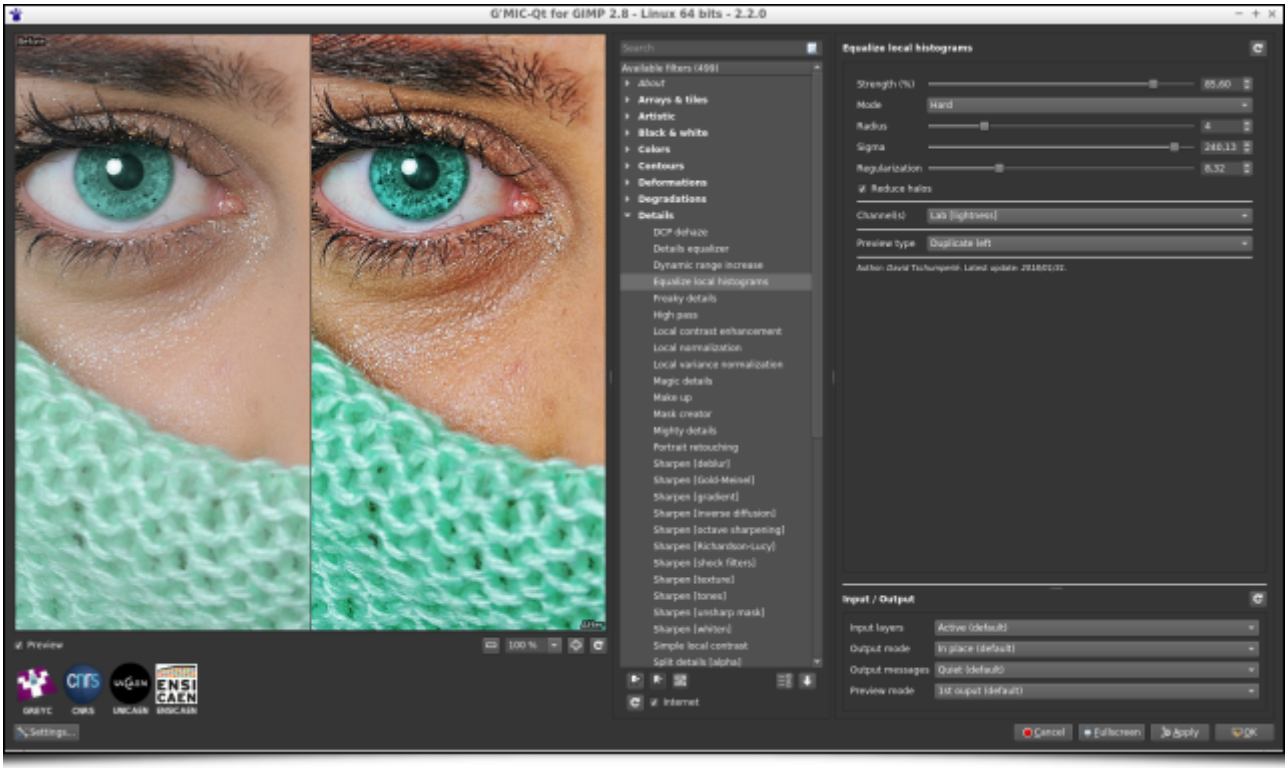


Fig. 3.2 : Les nouveaux filtres de rehaussement de détails de G'MIC.

Ainsi, l'application du nouveau filtre d'égalisation locale d'histogrammes de *G'MIC* sur l'image de paysage montagneux donne un résultat plus contrasté en détails géométriques et en couleurs, et ne fait plus apparaître de halos.

Fig. 3.3 : Différences de résultats entre le filtre Unsharp Mask *classique et l'égalisation locale d'histogrammes de G'MIC, pour le rehaussement de détails.*

3.2. Des déformations en tout genre

Régulièrement, de nouveaux filtres permettant de réaliser des déformations géométriques sur les images sont ajoutés à *G'MIC*, et cette nouvelle version majeure 2.2 ne déroge pas à cette règle.
Commençons donc par le filtre *Deformations / Spherize*, qui permet de déformer localement l'image pour donner l'impression que celle-ci est projetée sur une sphère ou un ellipsoïde 3D. C'est le filtre idéal pour transformer votre odieux collègue de bureau en **Monsieur Patate**[W] !

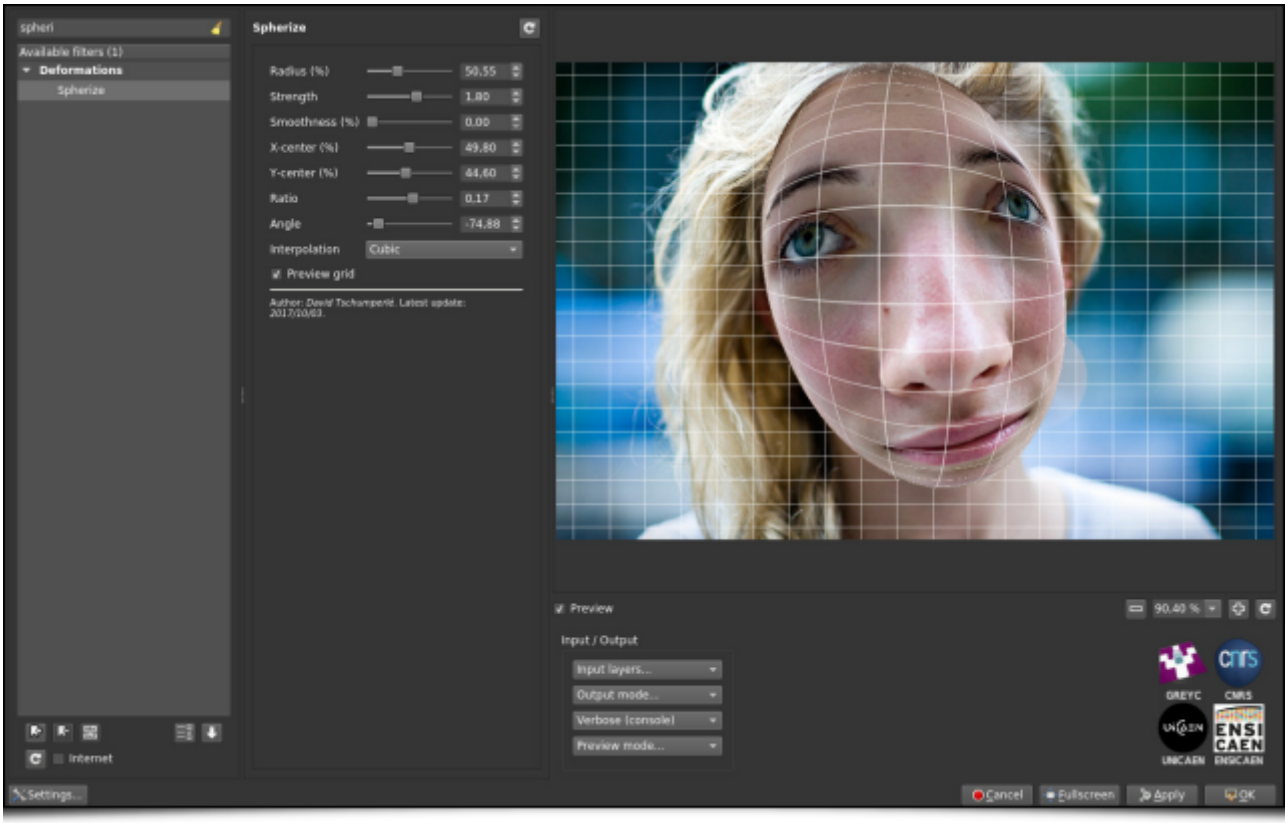

spheri
Available filters (1)
Deformations
Spherize
Spherize
Radius (%) 50.55
Strength 1.80
Smoothness (%) 0.00
X-center (%) 49.80
Y-center (%) 44.60
Ratio 0.17
Angle -74.88
Interpolation Cubic
Preview grid
Author: David Tschumperlé. Latest update: 2017/10/03.
Internet
Settings...
Preview
90.40 %
Input / Output
Input layers...
Output mode...
Verbose (console)
Preview mode...
GREYC
CNRS
UNICAEN
ENSICAEN
Cancel
Fullscreen
Apply
OK

Fig .3.4 : Deux exemples de déformations sphériques 3D réalisées avec le filtre Spherize *de G'MIC.*

Le filtre *Deformations/Square to circle*, quant à lui, implémente des transformations directes et inverses d'un domaine carré (ou d'un rectangle) vers un disque (ainsi que décrit mathématiquement sur ***cette page***), ce qui permet de générer ce type de déformations.

Fig. 3.5 : Transformations directes et inverses d'un carré vers un disque.

L'effet *Degradations / Streak* remplace une zone d'image masquée par l'utilisateur (remplie par une couleur uniforme) par une ou plusieurs copies d'une zone voisine. Il fonctionne principalement comme l'outil de clonage de GIMP mais évite à l'utilisateur de réaliser le remplissage de tout le masque à la main.

Fig. 3.6 : Le filtre Streak *recopie une partie de l'image dans une zone masquée définie par l'utilisateur.*

3.3. Abstractions artistiques

Vous allez me dire que les déformations c'est sympathique, mais qu'on désire parfois transformer une image de façon plus radicale. Tournons-nous donc maintenant vers les nouveaux effets qui transforment une image en une version plus abstraite (simplifiée ou redessinée). Ces filtres ont en commun une analyse de la géométrie des structures de l'image, suivie d'une étape de re-synthèse — avec plus ou moins de bonheur.

Par exemple, le filtre *Contours / Super-pixels* tente de regrouper localement les pixels de même couleur pour former une image partitionnée, façon puzzle, par des formes géométriques qui collent aux contours. Cette partition est obtenue via la méthode **SLIC** (*Simple Linear Iterative Clustering*), un algorithme classique de partitionnement d'images, qui a l'avantage d'être relativement rapide en temps de calcul.

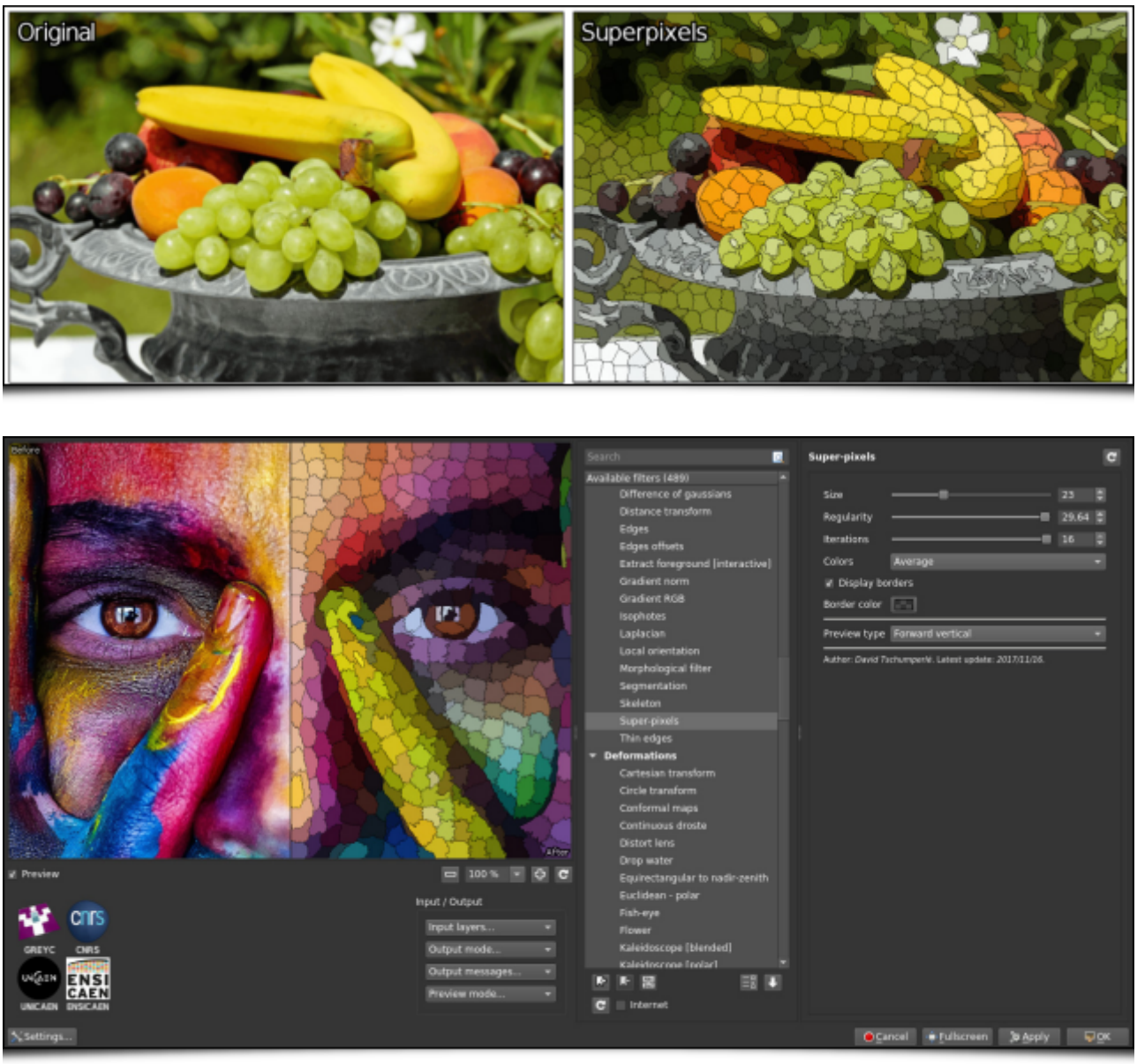
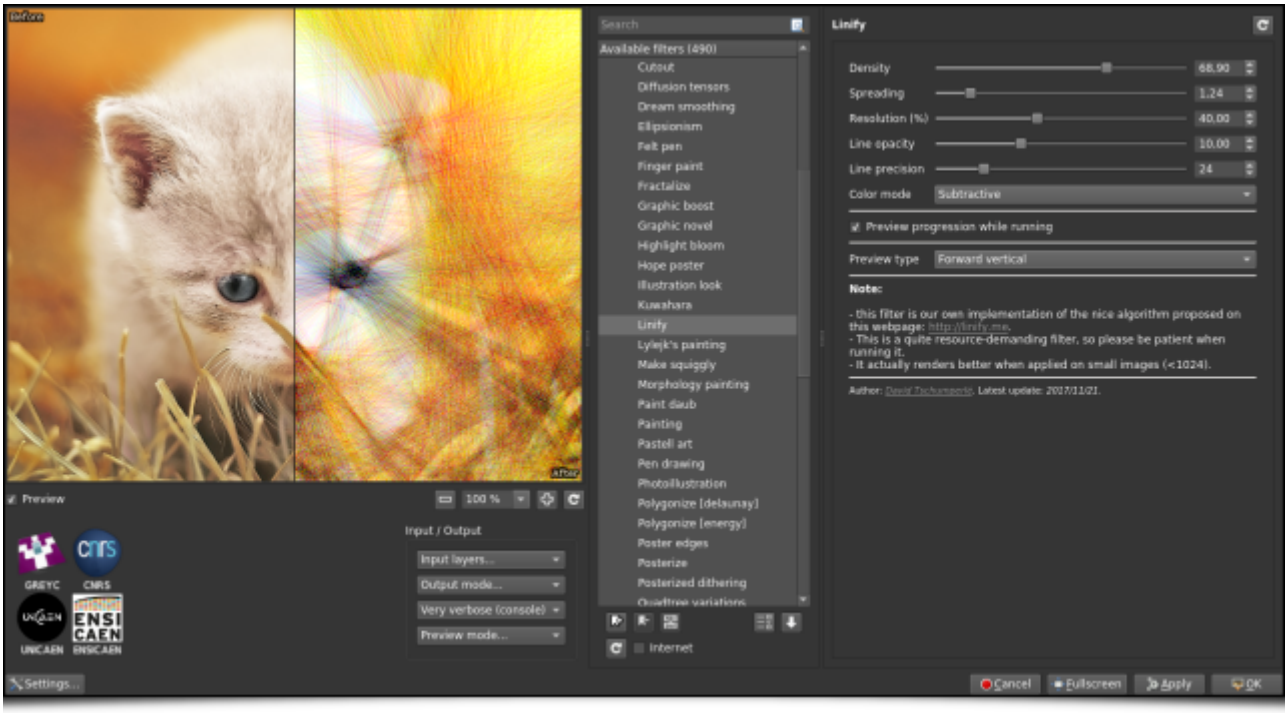


Fig. 3.7 : Décomposition d'une image en super-pixels, par l'algorithme SLIC (Simple Linear Iterative Clustering).

Le filtre *Artistic / Linify* tente quant à lui de redessiner une image d'entrée en superposant des droites colorées semi-transparentes sur un canevas initialement blanc, comme le montre la figure ci-dessous. Cet effet est la ré-implémentation d'un algorithme ingénieux initialement proposé sur le site **http://linify.me/** (implémenté en JavaScript).



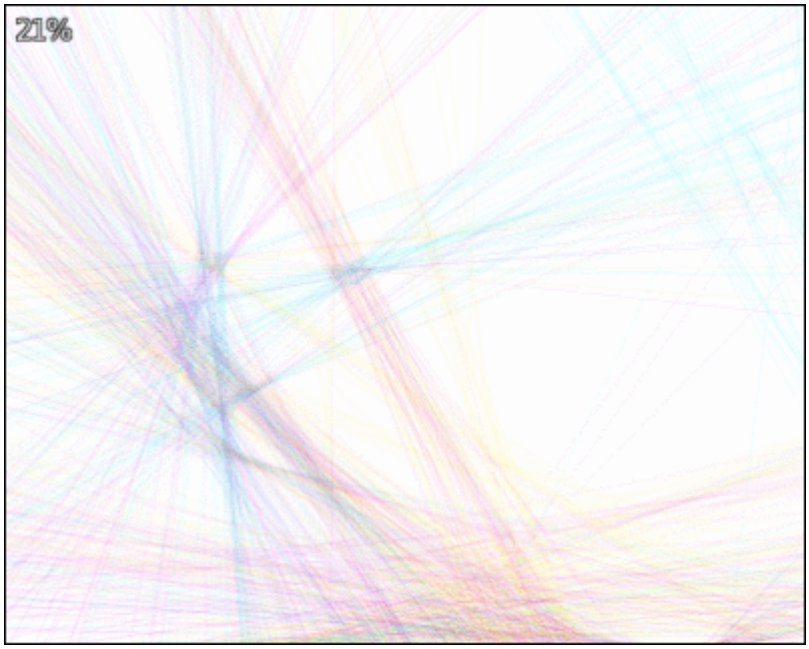


Fig. 3.8 : L'effet « Linify » cherche à redessiner une image en superposant uniquement des droites colorées semi-transparentes sur un canevas blanc.

L'effet *Artistic / Quadtree variations* va pour sa part décomposer une image dans un premier temps en ***quadtree***[W] (arbre quaternaire), puis la re-synthésiser en dessinant des ellipses orientées sur un canevas initialement vide, pour chaque feuille du *quadtree* ainsi obtenu, ce qui donne un petit effet « peinture » assez intéressant. Il est probable qu'avec des formes plus complexes que des ellipses pleines, on puisse synthétiser des rendus encore plus attrayants. Sûrement une idée à garder dans un coin du cerveau pour une prochaine mise à jour des filtres.

Fig. 3.9 : La décomposition d'image en quadtree permet par la suite de la re-synthétiser en superposant uniquement des ellipses orientées et colorées.

3.4. « Y en a un peu plus, je vous le mets quand même ? »

Et maintenant que vous avez traité pleins de belles images, pourquoi ne pas les exposer sous la forme d'un superbe montage photo ? C'est précisément le rôle du filtre *Arrays & tiles / Drawn montage*, qui permet de créer très rapidement une juxtaposition de photographies en épousant des formes quelconques. L'idée est de fournir au filtre un patron (*template*) coloré en plus de l'ensemble des photographies (*fig. 3.10a*), puis d'associer chaque photographie à chaque couleur différente du patron (*fig. 3.10b*). L'incrustation se fait alors automatiquement par *G'MIC*, qui redimensionne les images de telle sorte qu'elles apparaissent le mieux cadré possible à l'intérieur des formes définies par le patron (*fig. 3.10c*).
Nous avons réalisé **une vidéo tutorielle** montrant l'utilisation de ce filtre particulier.

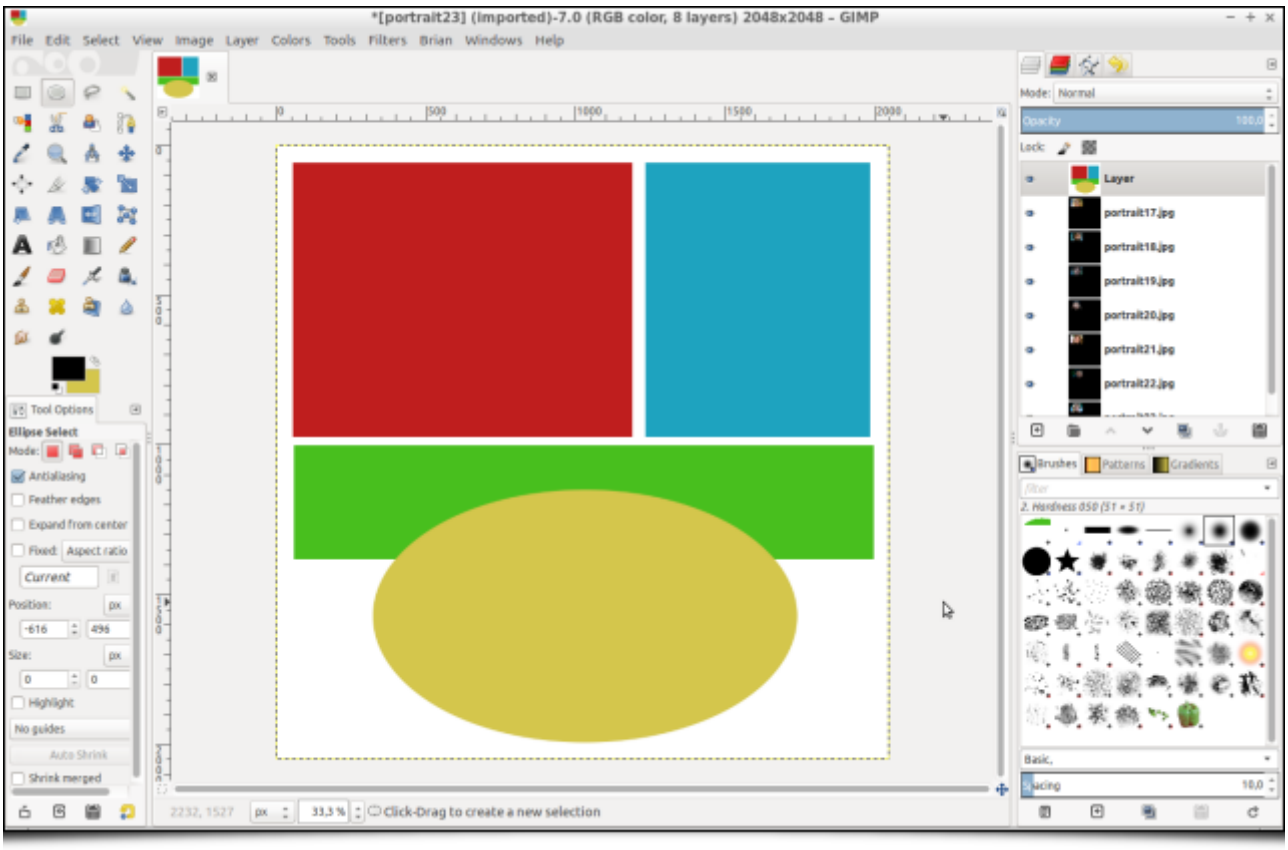


Fig. 3.10a : Étape 1 : l'utilisateur dessine l'organisation désirée du montage avec des formes de différentes couleurs.

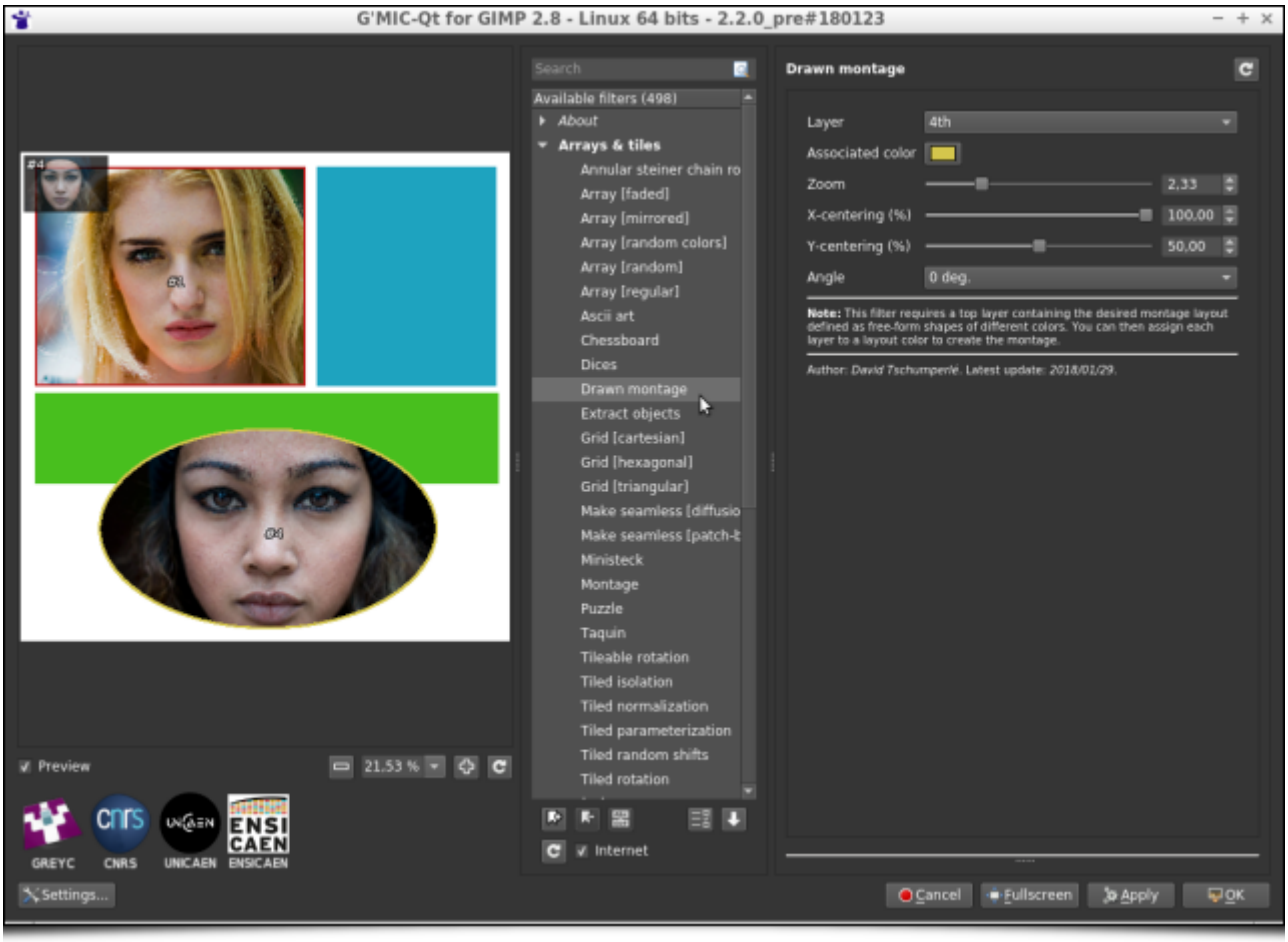


Fig. 3.10b : Étape 2 : le filtre Drawn montage *de G'MIC permet ensuite d'associer une photographie à chaque couleur.*

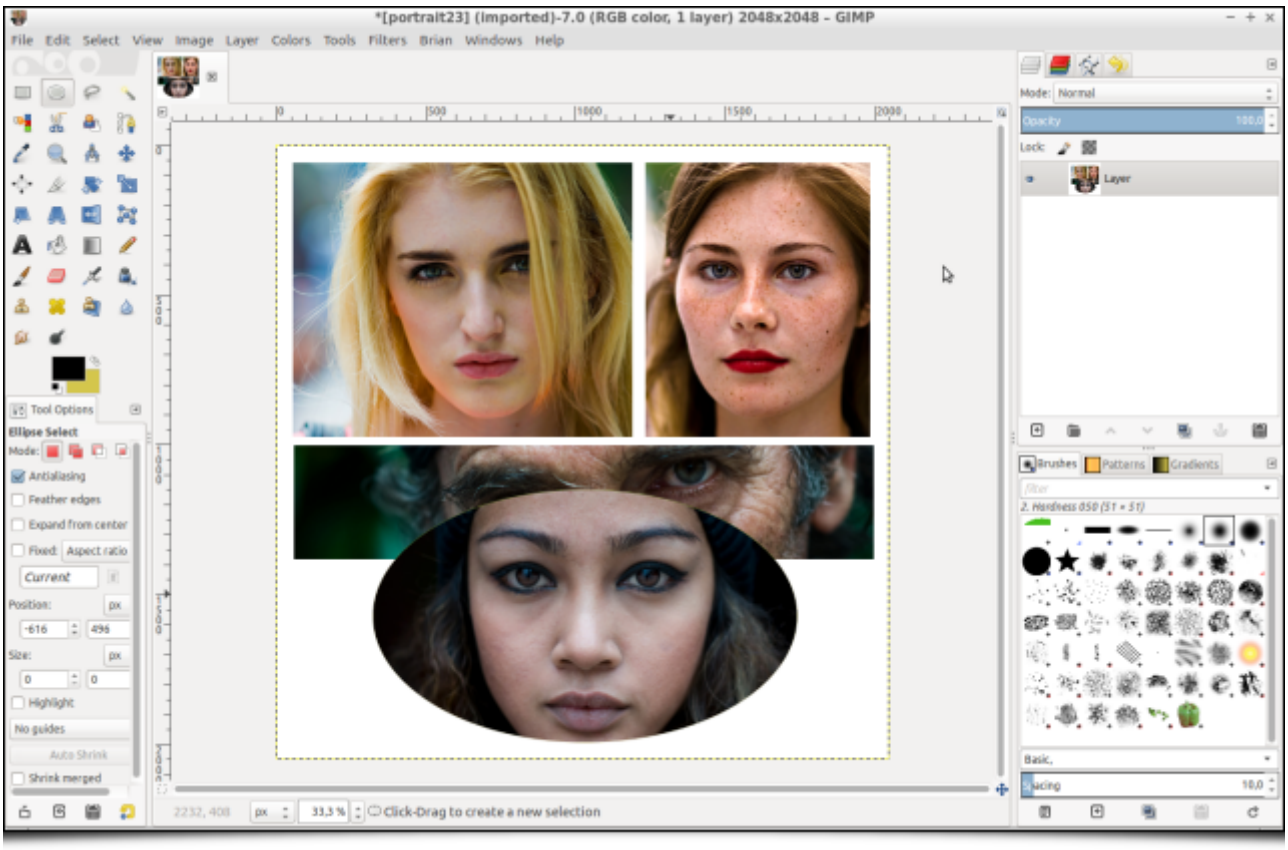


Fig. 3.10c : Étape 3 : le montage est finalement créé de manière automatique par le filtre, qui insère les photographies dans chaque forme de couleur en adaptant la résolution de chaque photographie.

Mais revenons en à des questions plus essentielles : avez vous déjà eu besoin de dessiner des rouages ? Non ?! C'est normal, ce n'est pas quelque chose que l'on fait tous les jours ! Mais, au cas où, le filtre *Rendering / Gear* se propose de le faire pour vous, avec différents réglages pour régler taille, couleurs et nombre de dents. Parfaitement inutile, donc totalement indispensable !

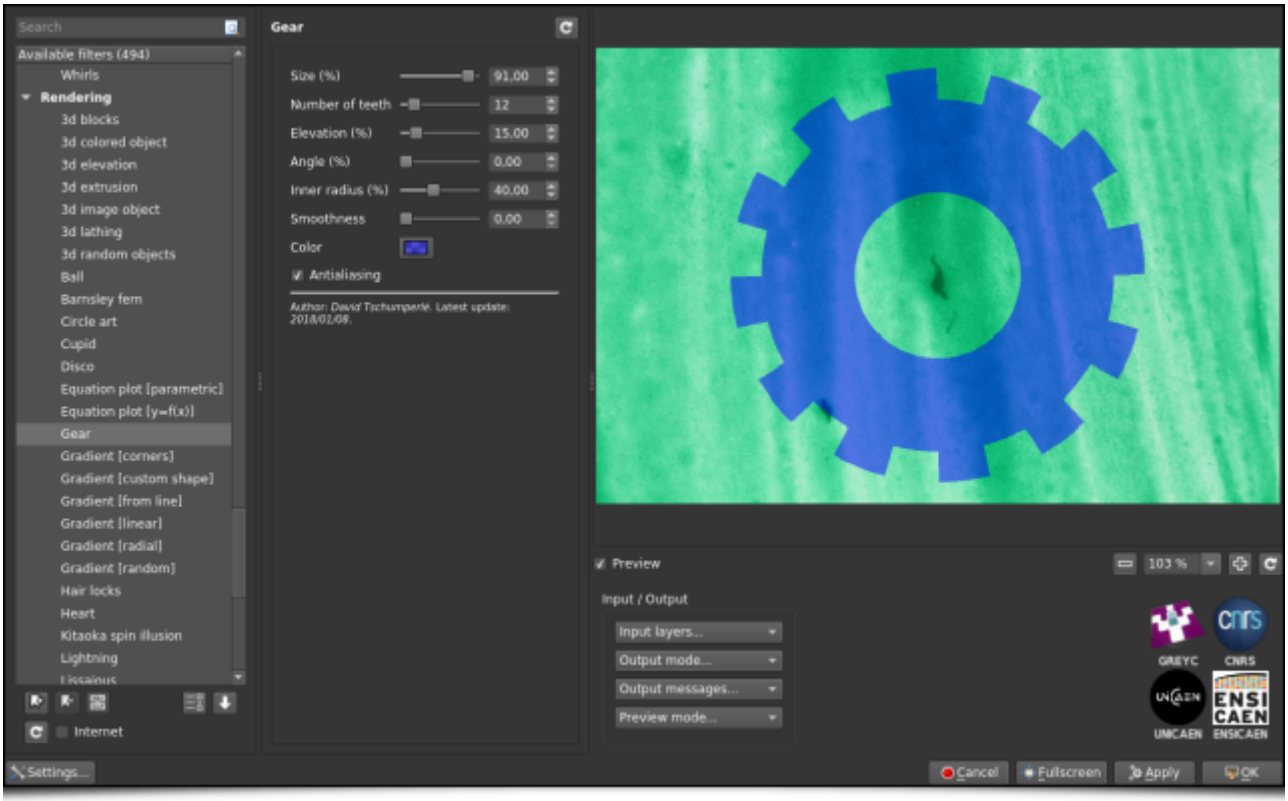


Fig. 3.11 : Le filtre Gear *tournant à plein régime.*

Besoin rapide d'une texture de satin ? Non plus ?! Dommage, car le filtre *Patterns / Satin* aurait pu vous être d'un grand secours !

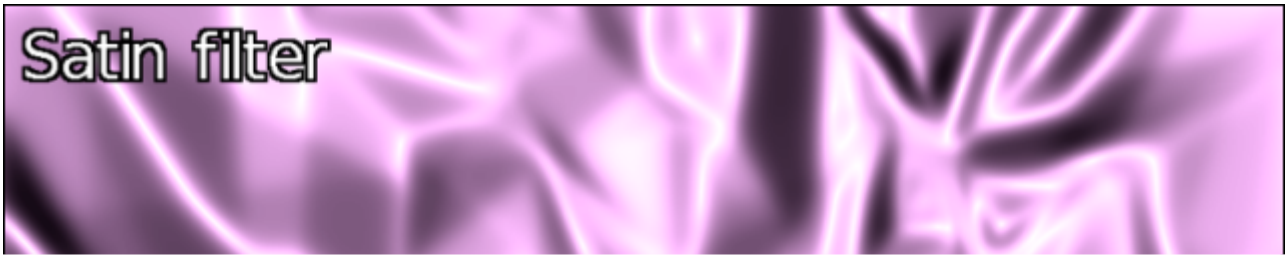
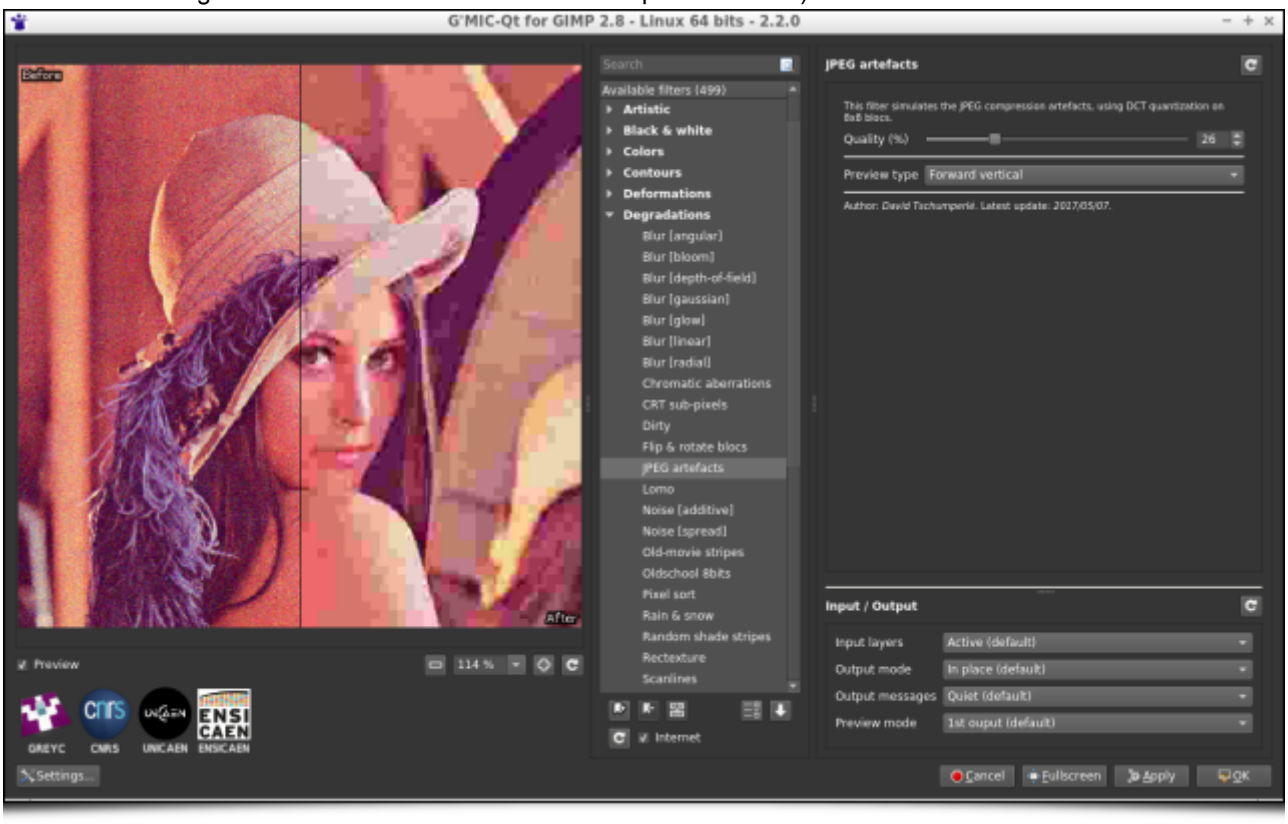


Fig. 3.12 : Le filtre Satin *de G'MIC vous rend la vie plus soyeuse.*

Et pour finir la série de ces « effets qui ne servent à rien sauf le jour où on en a besoin », notons la venue du filtre *Degradations / JPEG artefacts* qui simule l'apparition d'artefacts de compression JPEG due à la quantification des coefficients **DCT**[W] encodant les blocs 8 × 8 composant l'image (oui, vous obtiendrez quasiment la même chose en sauvant votre image sous forme d'un fichier JPEG de la qualité désirée).



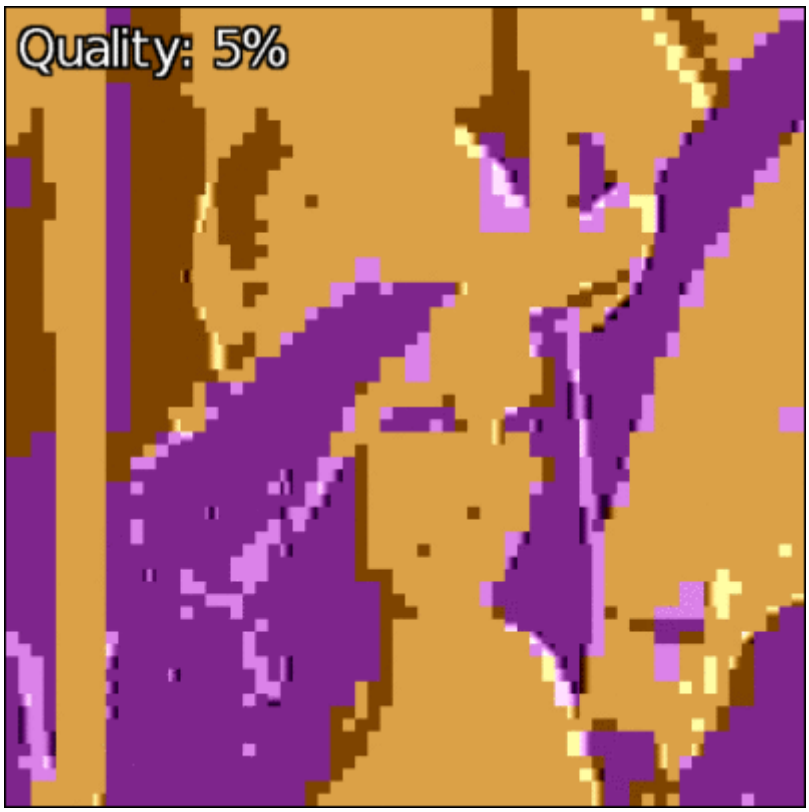


Fig. 3.13 : L'effet JPEG artefacts *simule une dégradation d'image due à la compression DCT par bloc 8 × 8.*

4. Autres améliorations notables

Cette revue des nouveaux filtres disponibles ne doit pas faire oublier les diverses améliorations qui ont également été apportées « sous le capot » et qui sont toutes aussi importantes, même si elles sont moins visibles en pratique.

4.1. Amélioration de l'interface du greffon G'MIC-Qt

Un gros effort de nettoyage et de restructuration du code du greffon *G'MIC-Qt* a été réalisé, avec pour conséquence pas mal de petits soucis réglés dans l'utilisation de l'interface graphique (***GUI*** [W]). Citons également en vrac quelques nouvelles fonctionnalités ayant fait leur apparition dans ce greffon : la possibilité de régler un délai d'expiration (***timeout*** [W]) lors de la prévisualisation du résultat de filtres un peu longs à calculer, une meilleure prise en compte des paramètres d'entrées-sorties pour chaque filtre (persistance, meilleure localisation, bouton de réinitialisation), des facilités pour maximiser la taille de la fenêtre de prévisualisation, pour éditer à la main son coefficient d'agrandissement (*zoom*), ou encore pour choisir la langue de l'interface (indépendamment de la langue du système), etc. Au final, plein de petites choses qui, additionnées, améliorent progressivement l'expérience utilisateur.

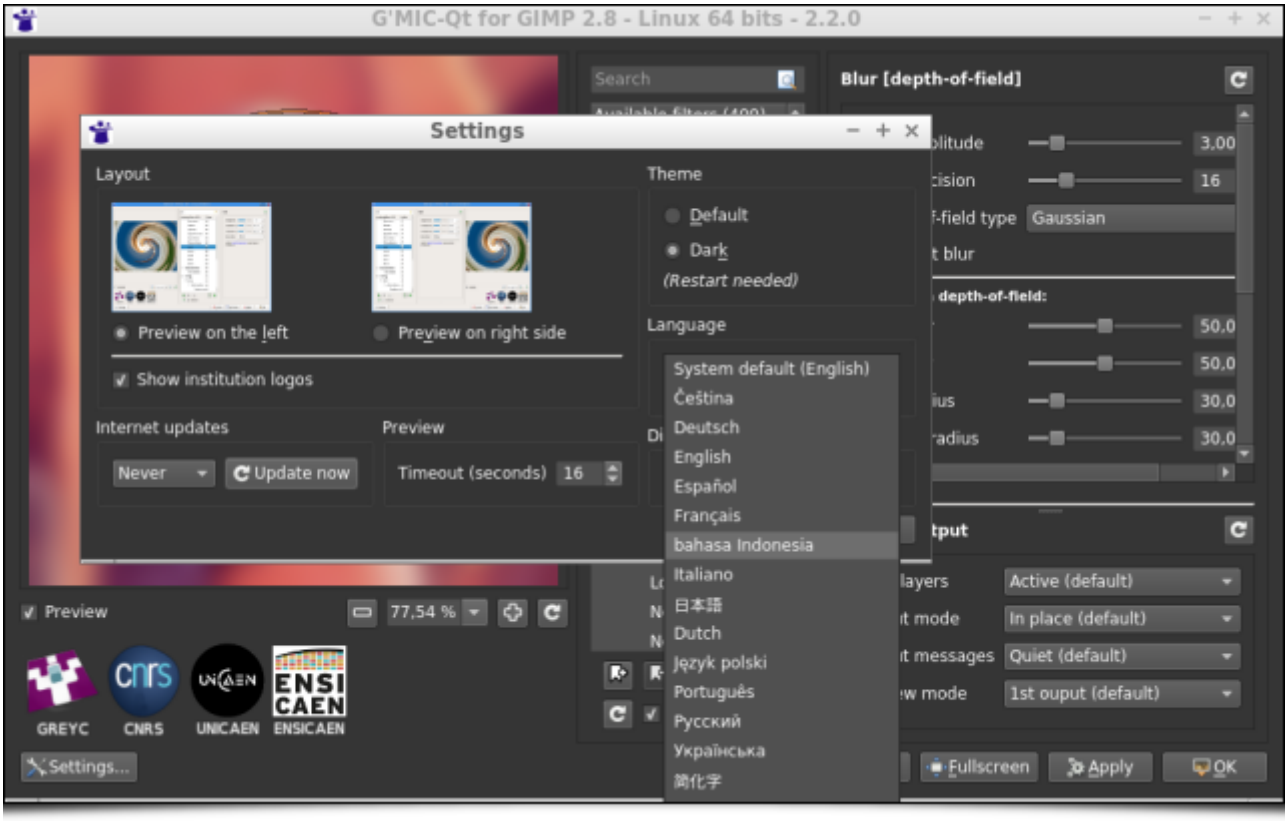


Fig. 4.1 : Aperçu de l'interface du greffon G'MIC-Qt dans sa dernière mouture 2.2.

4.2. Améliorations apportées au cœur de calcul

Encore moins visible, mais tout aussi important, beaucoup d'améliorations sont apparues dans le code du cœur de calcul de *G'MIC* et son interpréteur de langage de script associé. Il faut savoir que tous les filtres proposés dans le greffon sont écrits sous forme de scripts en langage *G'MIC*, et chaque petite amélioration apportée à l'interpréteur a donc une conséquence bénéfique pour l'ensemble des filtres proposés. Sans rentrer dans les détails techniques de ces améliorations internes, on peut citer pêle-mêle :

- des améliorations notables en ce qui concerne la syntaxe même du langage, allant de pair avec une meilleure performance d'analyse de cette syntaxe et, *in fine*, un temps d'exécution amélioré des scripts, pour une utilisation mémoire qui, elle, devient plus réduite ;
- l'évaluateur d'expressions mathématiques intégré à *G'MIC* connaît lui aussi quelques améliorations et nouvelles fonctionnalités, pour envisager encore plus de possibilités de codage d'algorithmes réalisant des opérations non triviales au niveau pixel ;
- une meilleure prise en charge des entrées-sorties de séquences vidéo brutes en format `.yuv` avec la prise en charge des formats `4:2:2` et `4:4:4`, en plus du format `4:2:0` qui était le seul géré auparavant ;
- enfin, deux nouvelles animations ont été ajoutées au menu des démonstrations de *G'MIC* (qui s'affiche par exemple lorsque l'on lance l'interface CLI — `gmic` sans arguments, depuis la ligne de commande) :

- tout d'abord, un effet de champ d'étoiles en 3D :

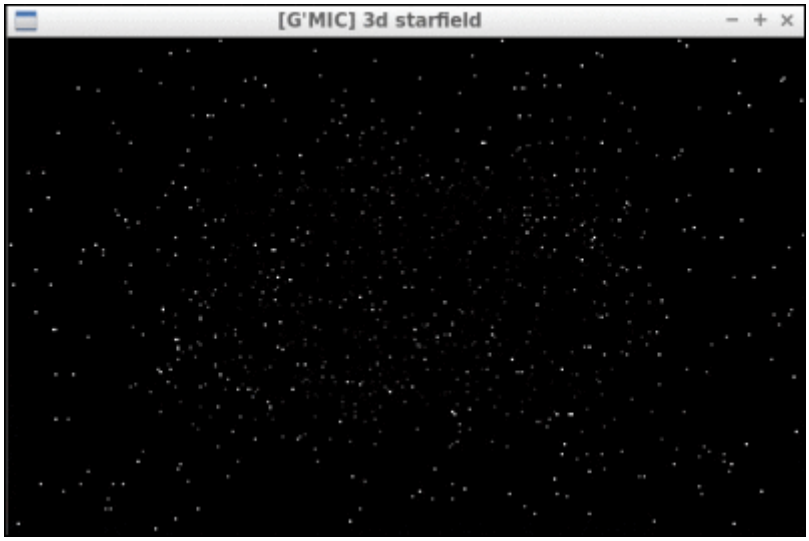


Fig. 4.2 : Nouvelle animation 3D Starfield *ajoutée au menu de démonstration.*

- mais aussi une version 3D jouable des ***Tours de Hanoï***[W] :

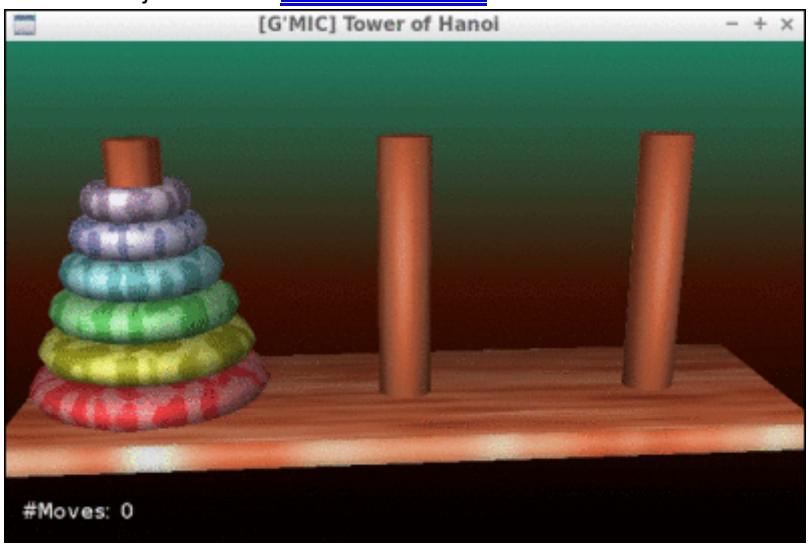


Fig. 4.3 : La version 3D jouable des « Tours de Hanoï », disponible dans G'MIC.

- pour finir, signalons l'apparition de la commande `tensors3d` dédiée à la représentation 3D d'un **champ tensoriel**[W] d'ordre 2. Et, en pratique, ça ne sert pas qu'à donner envie de manger des *Smarties*®! On peut par exemple l'utiliser pour visualiser certaines régions de volumes **IRM de tenseurs de**

diffusion [W] :

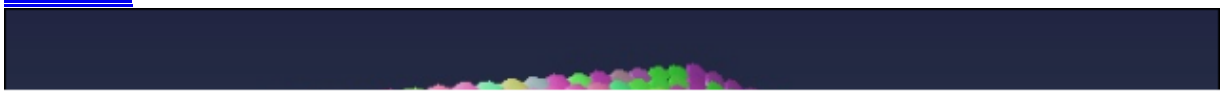


Fig. 4.4 : La commande tensors3d *permet la visualisation 3D de champs tensoriels d'ordre 2.*

4.3. Nouveau design pour *G'MIC Online*

Pour en finir avec le rayon des nouveautés, mentionnons également la refonte complète du service en ligne ***G'MIC Online*** durant l'année 2017, réalisée par Christophe Couronne et Véronique Robert du service développement du laboratoire GREYC.
G'MIC Online est un service Web, qui propose d'appliquer un sous-ensemble des filtres de *G'MIC* sur vos images, via un navigateur Web. Ces pages ont maintenant un **design adaptatif** [W], ce qui les rend utilisables de manière plus agréable qu'auparavant sur les appareils mobiles (*smartphones* et tablettes), comme illustré ci-dessous avec une copie d'écran du service lancé depuis un navigateur Chrome tournant sous Android, le tout sur une tablette de 10 pouces de diagonale.

Fig. 4.5 : Nouvelle version adaptative du service Web G'MIC Online, *tournant ici sur une tablette 10".*

5. Conclusion et perspectives

Voilà, le tour d'horizon de cette nouvelle version 2.2 de *G'MIC* est terminé. Une conclusion pourrait être : « Il y a plein de perspectives ! ». *G'MIC* est un projet libre que l'on peut qualifier de mature : les premières lignes de code ont été composées il y a presque dix ans, et l'on a aujourd'hui une bonne idée des possibilités (et des limites) de la bête. Nous espérons rencontrer toujours plus d'intérêt de la part des utilisateurs et développeurs libres, par exemple pour l'intégration du greffon générique *G'MIC-Qt* dans des logiciels divers et variés de traitement d'images ou de vidéos.
La possibilité d'utiliser le cœur de *G'MIC* sous une licence *CeCILL-C* — plus permissive — peut également être source de collaborations intéressantes dans le futur (certaines entreprises nous ont déjà approché à ce sujet).

Tout en étant à l'affût de collaborations potentielles, nous allons continuer à développer *G'MIC* dans la mesure de nos moyens et l'alimenter avec de nouveaux filtres et effets, au gré des suggestions de nos utilisateurs enthou-

siastes. Merci à eux pour leur aide et leurs encouragements constants (la motivation pour écrire du code ou des articles, passé 23 h, ne serait pas la même).

Et en attendant une prochaine dépêche sur *G'MIC*… « Vive le traitement d'images et la création artistique libre ! »

Aller plus loin

- **Le projet G'MIC** (814 clics)
- **Fil Twitter** (95 clics)
- **Le Web service G'MIC Online** (549 clics)
- **Série d'articles G'MIC sur LinuxFr.org** (536 clics)

Logiciel d'utilité publique

Posté par **Pierre Jarillon** (**site web personnel**) le 16/02/18 à 21:12. Évalué à 10. Dernière modification le 19/02/18 à 01:20.

> Logiciel d'utilité publique : voilà comment on devrait qualifier G'MIC !
> Ceci pour plusieurs raisons :
>
> - il utilise Qt, qui est porté sur toutes les grandes plates-formes et portable sur les autres ;
> - il est financé par nos impôts, fourni sous une licence libre, ce qui devrait servir de modèle à bien d'autres travaux universitaires ;
> - il est utilisé par GIMP et Krita, les deux principaux logiciels graphiques libres ; il peut et devrait être une nouvelle matière enseignée dans les écoles.
>
> C'est une suggestion pour la DINSIC (gestionnaire du RGI) : déclarer les logiciels libres majeurs d'intérêt public. Cela permettrait par exemple de financer LibreOffice et pourquoi pas une distribution. Mageia par exemple ?

Re: Logiciel d'utilité publique

Posté par **xavier philippon** le 17/02/18 à 07:49. Évalué à 6.

> Effectivement,
>
> Aujourd'hui, seuls les dons aux "Associations reconnues d'utilité publique ou d'intérêt général" sont déductibles.
> L'idée est donc de rendre partiellement déductibles les dons aux mainteneurs de logiciels libres "majeurs".
> Dans le détail,cela implique trois choses:
> **1.** Il existe une association dite "loi 1901" derrière le logiciel libre en question.
> **2.** L'association en question peut être reconnue "d'intérêt général". Selon moi l'utilité publique n'est pas appropriée.
> **3.** Un organisme lié à l'état lui a accordé ce statut.
> Pour les points 1 et 3, cela ne pose pas trop de problème. C'est pour le 2 que cela se complique. Est-ce que les logiciels libres peuvent prétendre à ce statut ?
> cela-dit, même dans la version actuelle de la loi, avec un bon dossier et quelques appuis bien choisis, ça doit pouvoir se faire.

Intégration dans Kdenlive

Posté par **vpinon** le 17/02/18 à 01:22. Évalué à 8.

> Hello,
>
> L'annonce de la 2.0 m'avait propulsé à écrire un filtre MLT appelant GMIC…
> Malheureusement je n'y ai passé que 2 soirées fatigué, freiné par une segfault par-ci, une corruption d'image par