

# G'MIC 1.6.2.0 : Colorisation de BD, transfert de couleurs, aide au détourage et autres réjouissances

Posté par [David Tschumperlé \(site web personnel\)](#) le 17/04/15 à 00:32. Édité par 7 contributeurs. Modéré par [Pierre Jarillon](#).  
Licence [CC By-SA](#).

Étiquettes : [gimp](#) , [krita](#) , [g'mic](#) , [traitement\\_d'images](#) + [Étiqueter](#)

À l'occasion de la sortie toute récente de la version **1.6.2.0** de [G'MIC](#) (*GREYC's Magic for Image Computing*), framework libre pour le traitement d'image, je vous propose de poursuivre [ma petite série d'articles de résumés](#) sur les avancées de ce logiciel libre sympathique, voire indispensable aux traiteurs d'images.

Cette dépêche présente un ensemble significatif des nouveautés ajoutées à G'MIC depuis la version **1.5.9.3** [présentée il y a environ 10 mois](#), à savoir l'apparition de nouveaux traitements liés à la couleur, l'aide au détourage avant-plan / arrière-plan, de nouveaux filtres artistiques et d'autres faits notables.

## Sommaire

- [1. Le projet G'MIC : contexte et présentation.](#)
- [2. Nouveaux traitements liés à la couleur.](#)
  - [Application de courbes dans des espaces couleurs quelconques.](#)
  - [Colorisation de BD / Comics.](#)
  - [Colorisation de photographies Noir & Blanc.](#)
  - [Transfert de couleurs.](#)
  - [Simulation de films argentiques.](#)
- [3. Aide à la décomposition avant-plan / arrière-plan.](#)
- [4. Nouveaux filtres artistiques.](#)
  - [Effet gravure.](#)
  - [Triangulation de Delaunay.](#)
  - [Autres filtres artistiques, en vrac.](#)
- [5. Autres faits notables.](#)
  - [Améliorations globales et bibliothèque \*libgmic\*.](#)
  - [Quelques nouveautés pour le filtrage d'images.](#)
  - [Amélioration du greffon G'MIC pour GIMP.](#)
  - [Débrider G'MIC : utilisation en ligne de commande.](#)
  - [Autres interfaces G'MIC en cours de développement.](#)
- [6. Perspectives et conclusions.](#)

## 1. Le projet G'MIC : contexte et présentation.



Fig. 1. Mascotte et logo du projet G'MIC, framework libre pour le traitement d'image.

Le projet *G'MIC* a vu le jour en Août 2008, dans l'équipe [IMAGE](#) du laboratoire [GREYC](#) (Unité Mixte de Recherche du [CNRS](#), située à Caen) et évolue depuis à un rythme soutenu. Il est distribué sous licence libre [CeCILL](#).

Il propose plusieurs interfaces utilisateurs pour la manipulation d'images génériques (images ou séquences d'images 2D ou 3D, multi-canaux, à valeurs flottantes, incluant donc les images couleurs classiques).

Son interface la plus populaire aujourd'hui est [un greffon](#) disponible pour le logiciel [GIMP](#).

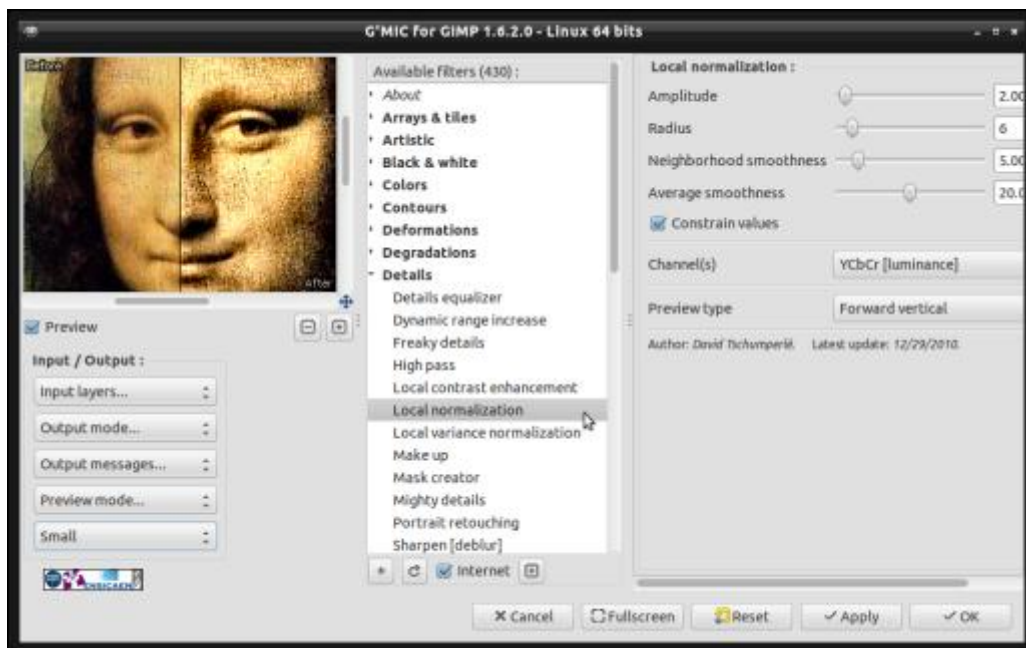


Fig. 1.1. Aperçu du greffon *G'MIC* pour *GIMP*.

Mais *G'MIC* peut aussi s'utiliser en [ligne de commande](#), de façon similaire à [ImageMagick](#) (pour des fonctionnalités optimales), ou directement via la page web [G'MIC Online](#). D'autres interfaces existent ([ZArt](#), un greffon pour [Krita](#), des greffons pour [Photoflow](#)...) mais celles-ci restent pour le moment plus confidentielles.

Toutes ces interfaces se basent sur la bibliothèque *libgmic*, écrite en C++, portable, thread-safe et multi-threadée (via l'utilisation d'[OpenMP](#) notamment), qui implémente toutes les fonctions de calculs proprement dites, et qui embarque son propre langage de script permettant aux utilisateurs avancés d'y ajouter leurs fonctions personnalisées de traitement d'image. Aujourd'hui, *G'MIC* comprend plus de [900 commandes](#) de traitement différentes, toutes paramétrables, pour une bibliothèque d'un peu moins de 5Mo. Les commandes que propose *G'MIC* couvrent un large spectre du traitement d'image, avec des algorithmes pour la manipulation géométrique, les changements colorimétriques, le filtrage (débruitage, rehaussement de détails par méthodes spectrales, variations, non-locales...), l'estimation de mouvement / le recalage, l'affichage de primitives, la détection de contours/la segmentation, le rendu d'objets 3d, le rendu artistique, etc.

C'est un ensemble d'outils très utiles, d'une part pour convertir, visualiser et explorer des données images, et d'autre part pour la construction de pipelines élaborés de traitements.

Il est entendu qu'étant le programmeur principal de ce logiciel, les avis que je vais exprimer ci-dessous sont totalement partiels et mon enthousiasme forcément débordant.

## 2. Nouveaux traitements liés à la couleur.

Voici quelques unes des nouvelles commandes et filtres qui ont été ajoutés dans *G'MIC* récemment, et qui concernent la création ou la modification de couleurs dans des images.

### Application de courbes dans des espaces couleurs quelconques.

Le concept de [courbes de couleurs](#)<sup>W</sup> est bien connu des artistes et photographes désirant retoucher les couleurs de leurs images. Basiquement, un outil de courbes de couleur permet d'appliquer à chaque composante *R,G,B* d'une image, une fonction 1d continue  $f:[0,255] \rightarrow [0,255]$  qui va transformer chaque valeur rouge, verte ou bleue  $x$  de chaque pixel (supposée initialement comprise entre 0 et 255) en la valeur  $f(x)$  donnée par la courbe

(fonction) définie par l'utilisateur (et dont les valeurs sont également comprises entre 0 et 255). Très souvent, ces fonctions sont construites par l'utilisateur, par la donnée de quelques points clés interpolés par des splines.

Mais que se passe-t-il si l'on veut appliquer une fonction de transformation des composantes couleurs dans un espace colorimétrique autre que *RGB* ? Et bien, on se brosse... Parce que la plupart des logiciels de retouche d'image ne permettent que l'application de courbes couleurs dans l'espace *RGB*.

Depuis peu, j'ai donc intégré dans *G'MIC* un outil de réglage de courbes couleurs dans des espaces colorimétriques autres que *RGB*, à savoir : [CMY<sup>w</sup>](#), [CMYK<sup>w</sup>](#), [HSI](#), [HSL](#), [HSV<sup>w</sup>](#), [Lab<sup>w</sup>](#), *LCh* ou [YCbCr<sup>w</sup>](#). C'est la commande `-x_color_curves` qui s'occupe de cette tâche (en ligne de commande), ou le filtre *Colors / Curves [interactive]* dans le greffon *G'MIC* pour *GIMP*. Un exemple d'utilisation est visible dans la figure ci-dessous (ici avec des courbes définies dans l'espace [Lab<sup>w</sup>](#)). Un [tutoriel vidéo](#) d'utilisation de ce filtre dans le greffon est également disponible. L'utilisation du filtre dans le greffon permet, par ailleurs, de sauver ses courbes favorites pour une application ultérieure sur un lot d'images par exemple.

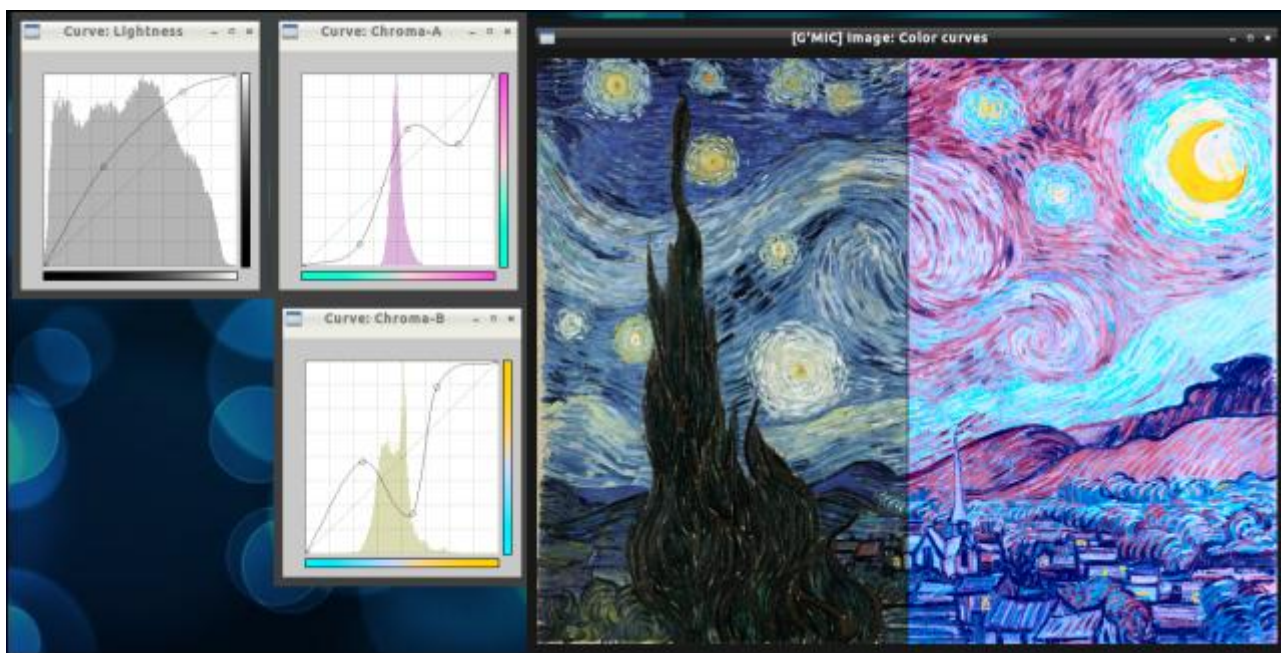


Fig.2.1. Définition interactive de courbes de couleurs dans l'espace colorimétrique **Lab** et application sur une image couleur. À gauche, les courbes définies par l'utilisateur pour les 3 composantes **L** (clarté), **a** et **b** (chrominances). À droite, le résultat de la transformation colorimétrique correspondante.

## Colorisation de BD / Comics.

C'est en discutant avec [David Revoy](#), illustrateur français de talent, et auteur entre autres du webcomics [Pepper & Carrot](#) (dont les moules de LinuxFr.org ont parlé dans [ce journal](#), ou encore [celui-ci](#)), qu'a germé l'idée d'un filtre qui pourrait aider les dessinateurs de BD à **coloriser** leurs images. Le travail de colorisation d'une BD est en effet quelque chose de long et de contraignant, même sur ordinateur, comme l'explique [David Revoy](#) sur [son blog](#).

Il existe déjà des outils d'aide à la colorisation, mais ceux-ci sont fermés, et il y avait donc un manque de ce côté-là pour les utilisateurs de logiciels libres.

J'ai donc pris mon courage (plutôt mon clavier) à deux mains, et j'ai implémenté dans *G'MIC* un filtre d'interpolation de couleurs « intelligent » (toute mesure gardée) qui essaye de générer un calque complet de couleurs, à partir seulement d'un nuage de points colorés. L'idée est de demander à l'illustrateur de positionner seulement quelques points-clés colorés sur son image (par exemple à l'intérieur des différentes zones), et de laisser l'algorithme en déduire une colorisation probable de tout le dessin (en analysant les contours et les objets qu'il contient). Avec [David Revoy](#) et [Timothée Giet](#) nous avons échangé via irc sur la façon d'avoir une interface de colorisation la plus intuitive possible, et le résultat, c'est une nouvelle commande *G'MIC* `-x_colorize` (utilisable en ligne de commande) et le filtre correspondant *Black & White / Colorize [interactive]* pour le greffon *G'MIC* de *GIMP*. Et apparemment, ça ne marche pas trop mal !

Il semblerait même que cet unique algorithme ait suffisamment intéressé les utilisateurs de *Krita* pour qu'il motive les développeurs à retravailler sur leur propre greffon *G'MIC* pour *Krita* pour inclure cet algorithme dedans. Depuis, David Revoy a utilisé cet algorithme régulièrement pour ses créations (notamment pour *Pepper & Carrot*). Ci-dessous, vous pouvez voir un exemple d'une phase de colorisation d'une case de BD de *Pepper & Carrot* initialement en niveau de gris, avec l'ajout des points-clés colorés et le résultat de la colorisation réalisée par l'algorithme de *G'MIC* (tout ceci réalisé avec le greffon sous *Krita*).

Sur [son site](http://www.davidrevoy.com), David a posté des vidéos détaillant tout le processus de colorisation qu'il utilise, et dont *G'MIC* fait maintenant partie.



Fig.2.2. Utilisation de *G'MIC* pour la colorisation de BD : Étape 1, ouverture du dessin à coloriser, ici sous *Krita*. (Cette image provient du site web de David Revoy : <http://www.davidrevoy.com/article240/gmic-line-art-colorization>)



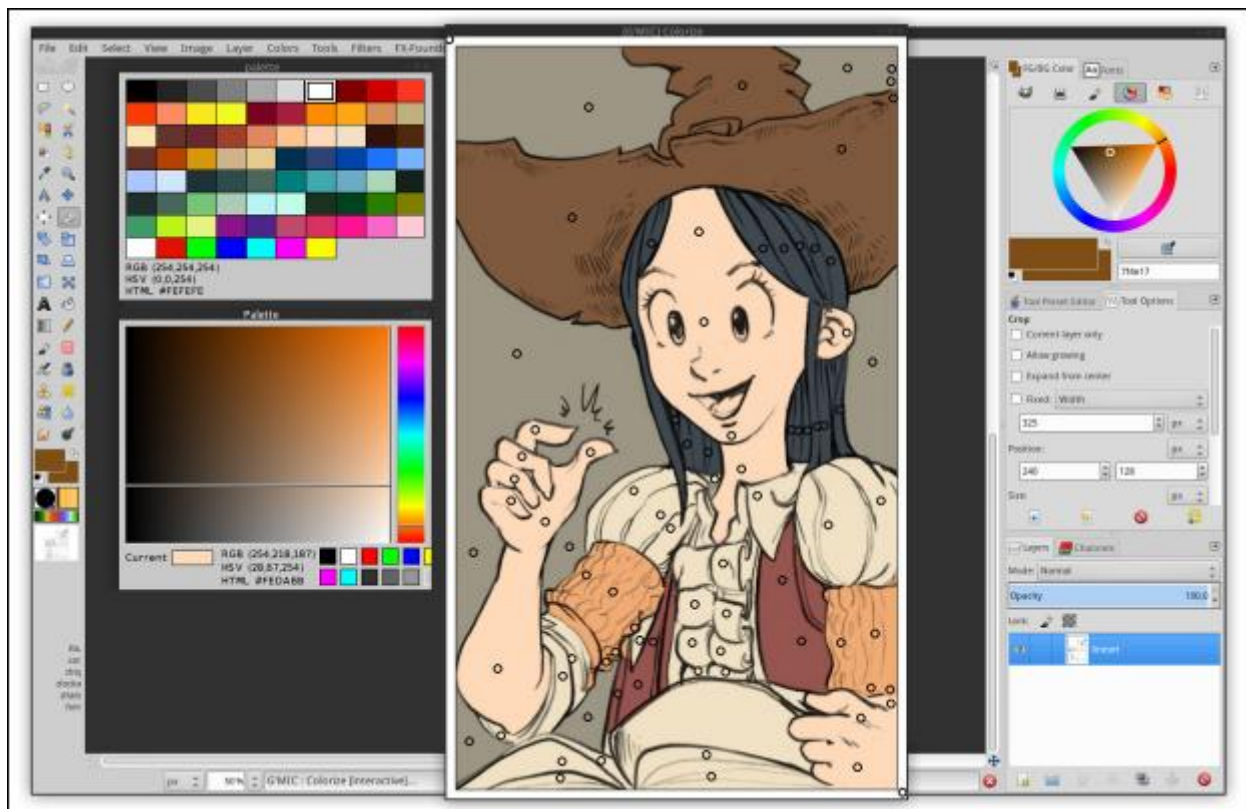


Fig.2.3. Utilisation de G'MIC pour la colorisation de BD. Étape 2, mise en place de quelques points-clés couleurs et colorisation automatique obtenue par G'MIC.

(Cette image provient du site web de David Revoy : <http://www.davidrevoy.com/article240/gmic-line-art-colorization>)

Après cette première étape de colorisation automatique, l'artiste peut ajouter à sa guise ombres et lumières sur les aplats couleurs générés, pour finaliser son œuvre. À noter que cette dernière étape est facilitée car chaque couleur est, de fait, facilement sélectionnable de manière séparée sur le calque couleur généré par l'algorithme. Et, pour ceux qui préfèrent travailler sur des calques de couleurs séparés plutôt que sur des sélections, le filtre de colorisation propose aussi la possibilité de générer  $N$  calques couleurs de sortie, chacun contenant une couleur unique.

Encore un grand merci donc à [David Revoy](#) et [Timothée Giet](#) pour leurs retours enthousiastes et pédagogiques, nous disposons donc maintenant d'un outil de colorisation de BDs *libre* que mon dentier nous envie :) Voilà un exemple de collaboration entre recherche académique et artistes "libres" qui a porté ses fruits.

## Colorisation de photographies Noir & Blanc.

Mais ce qu'on peut faire avec des BDs, peut-on le faire avec de vieilles photographies en noir et blanc, me demanderez-vous ? Hé bien oui ! En modifiant légèrement l'algorithme, nous pouvons de la même manière reconstruire la chrominance d'une photographie initialement en niveaux de gris, pour la coloriser à partir de points-clés couleurs définis par l'utilisateur. C'est ce qu'illustre l'exemple ci-dessous, avec la colorisation d'une vieille photo d'indien. À noter que le contenu de l'image étant sensiblement plus complexe qu'une case de BD, il est souvent nécessaire de placer beaucoup plus de points-clés pour coloriser une photographie. C'est le même filtre que précédemment, à savoir *Black & white / Colorize [interactive]* dans le greffon G'MIC pour GIMP, qui permet cela.



Fig.2.4. Colorisation d'une vieille photo Noir & Blanc avec l'algorithme de colorisation de G'MIC. (Ugh, vieil indien, excuse-moi pour la couleur improbable de tes cheveux, qui te fait ressembler à [Kim Kardashian](#)...)

## Transfert de couleurs.

Ici, le concept de « transfert de couleurs » correspond au fait de vouloir modifier les couleurs d'une image *A* en les remplaçant par les couleurs présentes dans une autre image *B* (l'image de *référence*), de telle sorte que l'image modifiée *A'* ait la même « ambiance colorimétrique » que l'image référence *B*, et ceci bien évidemment, de manière complètement automatique.

C'est un problème mal posé, complexe à résoudre, et il y a déjà pas mal d'articles de recherche parus à ce sujet ([ce papier](#) par exemple). La difficulté c'est d'arriver à générer une image *A'* en gardant un aspect « naturel », sans création d'aplats de couleurs ou au contraire sans apparition de discontinuités qui ne seraient pas présents initialement dans l'image *A* originale. Bref, ce n'est pas trivial.

Récemment, un algorithme de transfert de couleurs a été implémenté et intégré dans *G'MIC*, et il fonctionne de façon plutôt satisfaisante. Ce n'est pas parfait, certes, mais c'est un bon début. Le filtre *Colors / Transfer colors [advanced]* du greffon *G'MIC* pour *GIMP* permet d'appliquer facilement cet algorithme de transfert. C'est un filtre qui nécessite plusieurs calques d'entrée, dont un calque contenant l'image avec les couleurs de référence à transférer sur les images des autres calques. Voici comment cela se présente dans le greffon. On y voit l'image originale (à gauche dans la fenêtre de prévisualisation), l'image contenant les couleurs de référence (en bas à gauche), et l'image résultat (à droite).

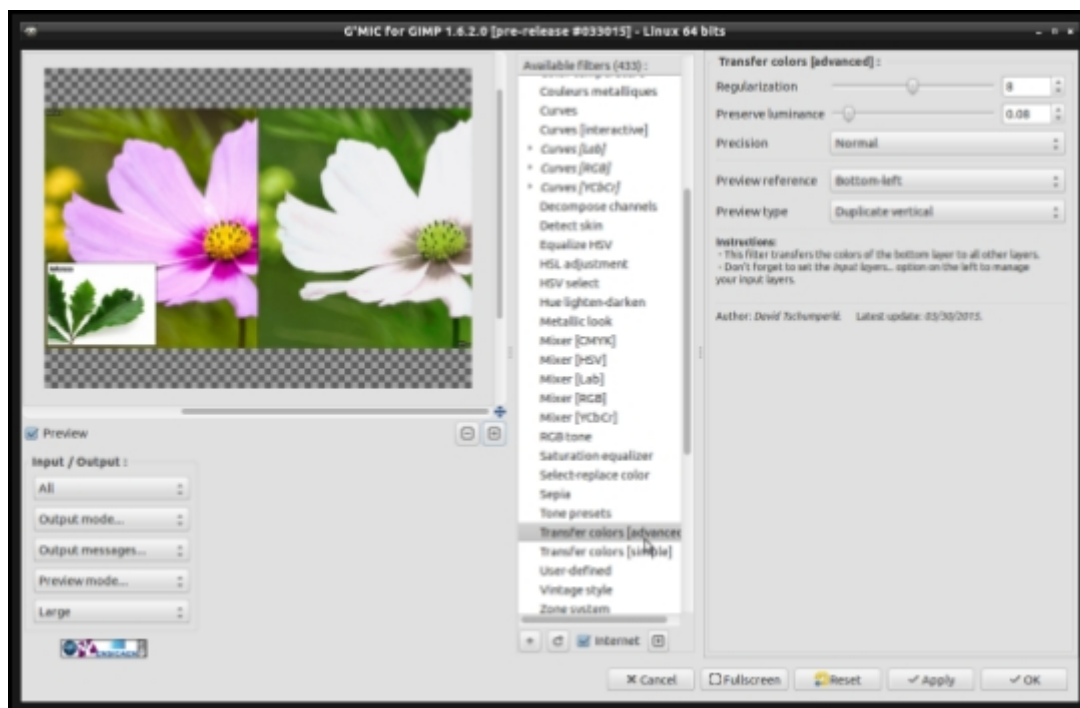


Fig.2.5. Aperçu du filtre de transfert de couleurs dans le greffon G'MIC pour GIMP.

Ce filtre réalise parfois des merveilles avec certaines paires d'images ! Deux exemples de transfert sont illustrés sur la figure ci-dessous. Bien sûr n'espérez pas de résultats miraculeux avec des cas pathologiques (transfert d'une image avec plein de couleurs sur une image monochrome par exemple) ! Mais globalement, ça fonctionne pas mal.



Fig.2.6. Deux exemples de résultats de transfert de couleurs d'une photo de référence (image du milieu) vers une photo source (image de gauche). Les images de droite sont les résultats générés par l'algorithme de transfert de G'MIC.

Il existe également un [tutoriel vidéo](#) qui illustre comment faire fonctionner ce filtre dans le greffon G'MIC pour GIMP.

On peut également penser à d'autres applications plus « subtiles » de ce filtre, par exemple pour l'homogénéisation de couleurs entre les *frames* ou les plans successifs d'une même vidéo, ou d'une paire d'images stéréoscopiques, par exemple.

Simulation de films argentiques.



[Pat David](#) est un photographe amateur américain très actif dans la communauté du logiciel libre orienté photographie, de par ses tutoriels et son expérience qu'il partage sur son blog. J'ai eu la chance d'interagir avec lui à de nombreuses occasions (dont une rencontre en live à [LGM'2014](#) à Leipzig), et cela a donné naissance à pas mal de filtres différents implémentés dans *G'MIC*. En particulier, nous avons proposé des filtres de simulation de films argentiques, comme ceux présentés ci-dessous. C'est seulement un petit échantillon des plus de 300 transformations couleurs qui ont été concoctées par *Patrick* et que nous avons rendu facilement accessibles.



Fig.2.7. Aperçu de quelques résultats de simulation de films argentiques, disponibles dans *G'MIC*.

(Cette image provient du site web de Pat David : <http://blog.patdavid.net/2013/08/film-emulation-presets-in-gmic-gimp.html>)

Ces filtres ont assez vite suscité un certain intérêt dans la communauté libre de retouche photo, et ont par la suite été repris et ré-implémentés dans le logiciel [RawTherapee](#).

Pour rendre ces filtres encore plus visibles, [une page web dédiée](#) a été mise au point, qui recense l'ensemble des transformations couleurs proposées et propose de télécharger les [CLUTs](#)<sup>W</sup> (fonctions de transfert) utilisées pour chaque transformation. À noter que vous pouvez retrouver ces filtres également sur [G'MIC Online](#) pour les tester directement sur vos propres images, directement via votre navigateur web.

Là encore, c'est une fonctionnalité qui manquait dans le logiciel libre (une sorte d'équivalent du logiciel propriétaire [DXO FilmPack](#)), et qui a pu être comblée assez rapidement via l'infrastructure proposée par *G'MIC*.

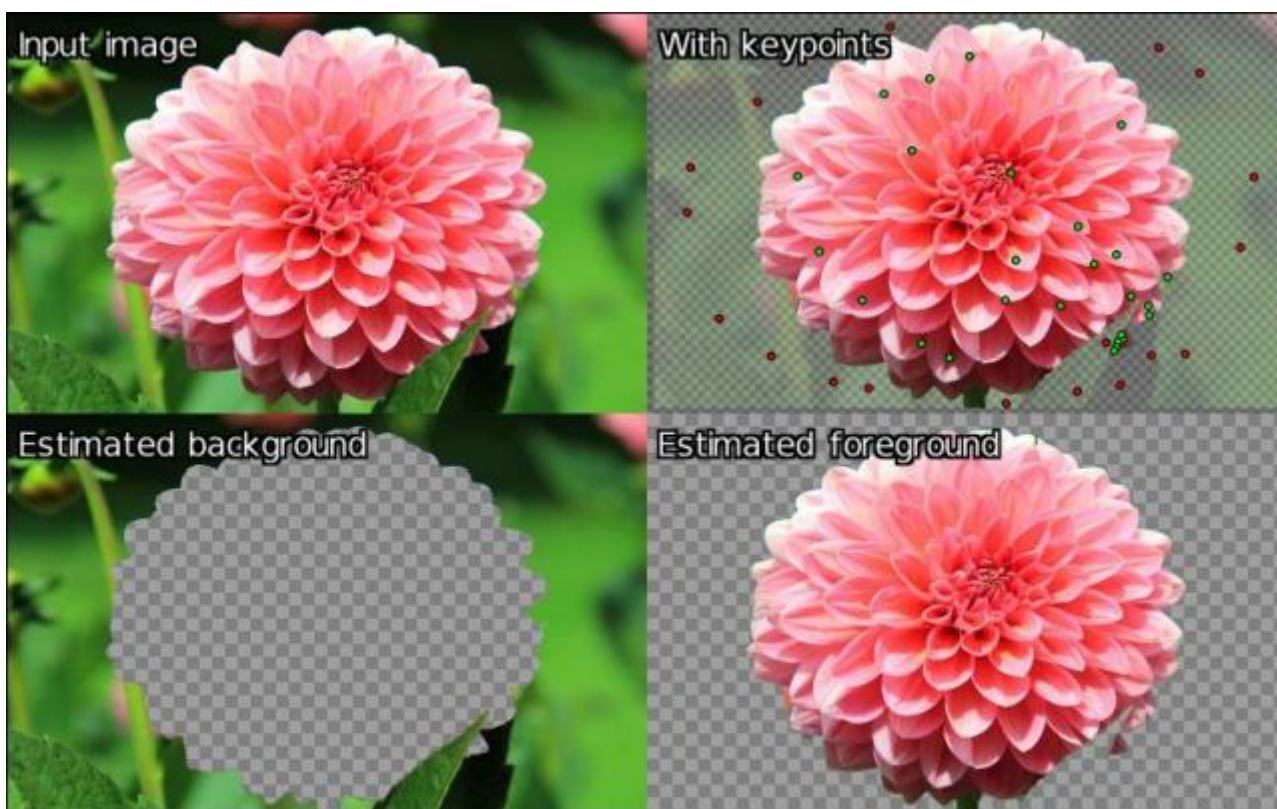
### 3. Aide à la décomposition avant-plan / arrière-plan.



Quand on fait de la retouche photo, il est assez courant que l'on veuille traiter de manière séparée les objets présents à l'avant-plan et dans le fond. Pour cela, on a souvent besoin de faire une décomposition d'image en avant-plan/arrière plan, en effectuant un détourage (avec l'outil de sélection "lasso" par exemple). Pour des objets un peu complexes, réaliser cette sélection est quelque chose de fastidieux. *G'MIC* incorpore maintenant un filtre permettant d'accélérer ce travail de découpe des objets. C'est la commande `-x_segment` (en ligne de commande) ou le filtre *Contours / Extract foreground [interactive]* du greffon *G'MIC* pour *GIMP* qui permet de réaliser cette tâche.

Le principe est exactement le même que pour le filtre de colorisation présenté précédemment : L'utilisateur place des points-clés, ayant pour label soit « avant-plan », soit « arrière-plan », et l'algorithme interpole ces labels en prenant en compte les contours présents dans l'image, et en déduit ainsi une carte binaire de détourage. Le filtre décompose ensuite l'image en deux calques distincts, un calque contenant les points d'avant-plan uniquement, et un autre avec les points d'arrière-plan. La figure ci-dessous présente un exemple de détourage effectué en utilisant ce filtre : L'utilisateur place quelques points-clés sur une image originale d'une fleur (image en haut à gauche). Les points verts correspondent aux objets d'avant-plan, tandis que les rouges sont placés dans le fond (image en haut à droite). À partir de ces données très éparées, et de l'analyse des contours de l'image, le filtre détermine automatiquement pour chaque pixel son appartenance au calque d'arrière-plan (image en bas à gauche) ou d'avant-plan (image en bas à droite).

Effectuer ce détourage ne prend ici que quelques secondes, alors que la même opération effectuée manuellement serait beaucoup plus longue (le contour de la fleur n'ayant pas une forme simple). Un [tutoriel vidéo](#) illustrant l'utilisation de ce filtre dans le greffon *G'MIC* pour *GIMP* est disponible.



*Fig.3.1. Décomposition avant-plan / arrière plan avec le filtre « Contours / Extract foreground » de G'MIC.*

Cette décomposition de l'image d'origine permet par la suite d'appliquer des traitements différenciés sur l'avant-plan et sur l'arrière-plan. Par exemple ci-dessous nous modifions la teinte et la saturation du calque d'avant-plan uniquement, pour jouer sur la couleur de la fleur, sans modifier la couleur de l'arrière-plan. Le détourage d'objet étant un besoin très fréquent en retouche photo, je vous laisse imaginer le nombre d'utilisations possibles de ce filtre.



Fig.3.2. Après modification de la teinte et de la saturation des couleurs de l'avant-plan uniquement.

## 4. Nouveaux filtres artistiques.

G'MIC, c'est aussi tout une floppée de filtres amusants pour créer des effets artistiques sur vos images. En voici quelques-uns notables qui ont été ajoutés dernièrement.

### Effet gravure.

Le filtre *Black & White / Engrave* tente de transformer une image en gravure. Le nombre relativement élevé de paramètres permet d'obtenir beaucoup de contrôle sur le type de rendu obtenu.

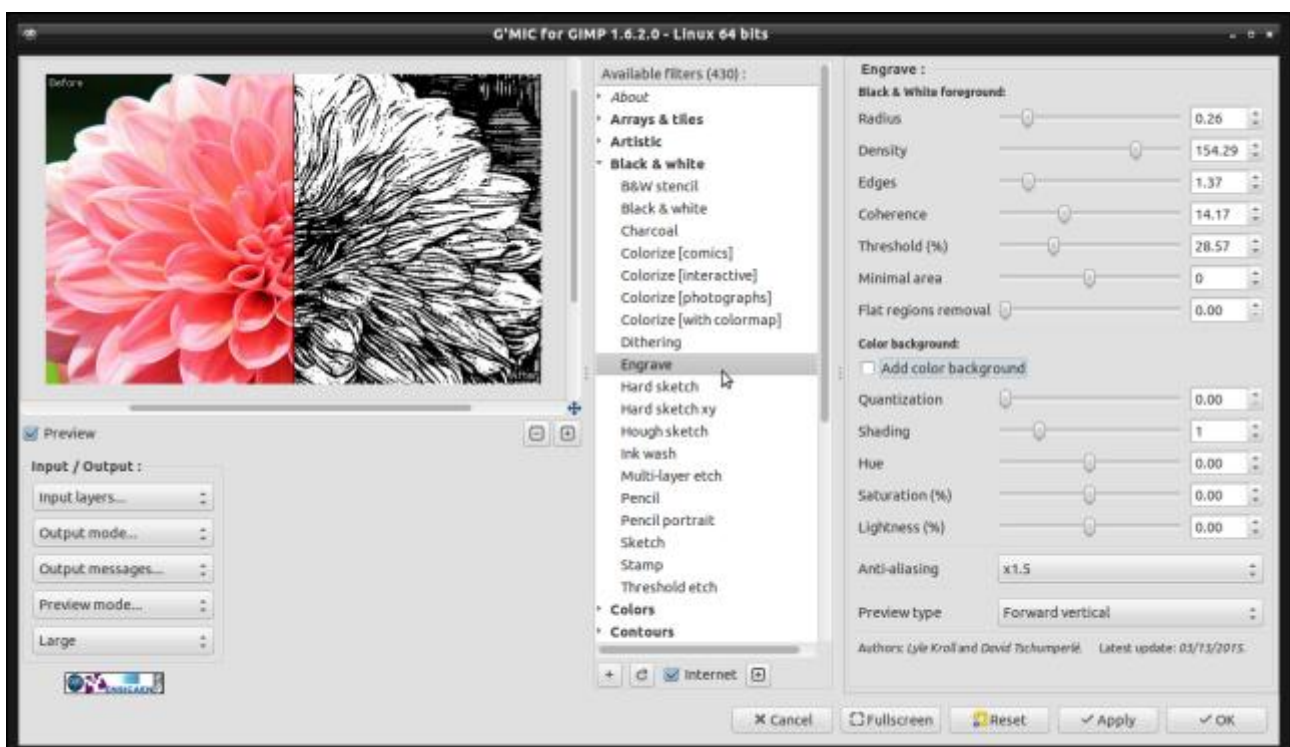


Fig.4.1. Aperçu du filtre « Gravure » dans le greffon G'MIC pour GIMP.



Là où ce filtre est particulièrement intéressant, c'est qu'il permet (en choisissant des paramètres adaptés) de transformer une image en version *Comics*, comme illustré avec les deux exemples ci-dessous. Un [tutoriel vidéo](#) existe également pour montrer les différentes étapes nécessaires à l'obtention de ce type de résultats, qui ne prend au final que quelques secondes (minutes pour les plus lents).



Fig.4.2. Conversion de photo (en haut à gauche) sous forme de dessin façon *Comics*, avec le filtre "Engrave" de G'MIC.

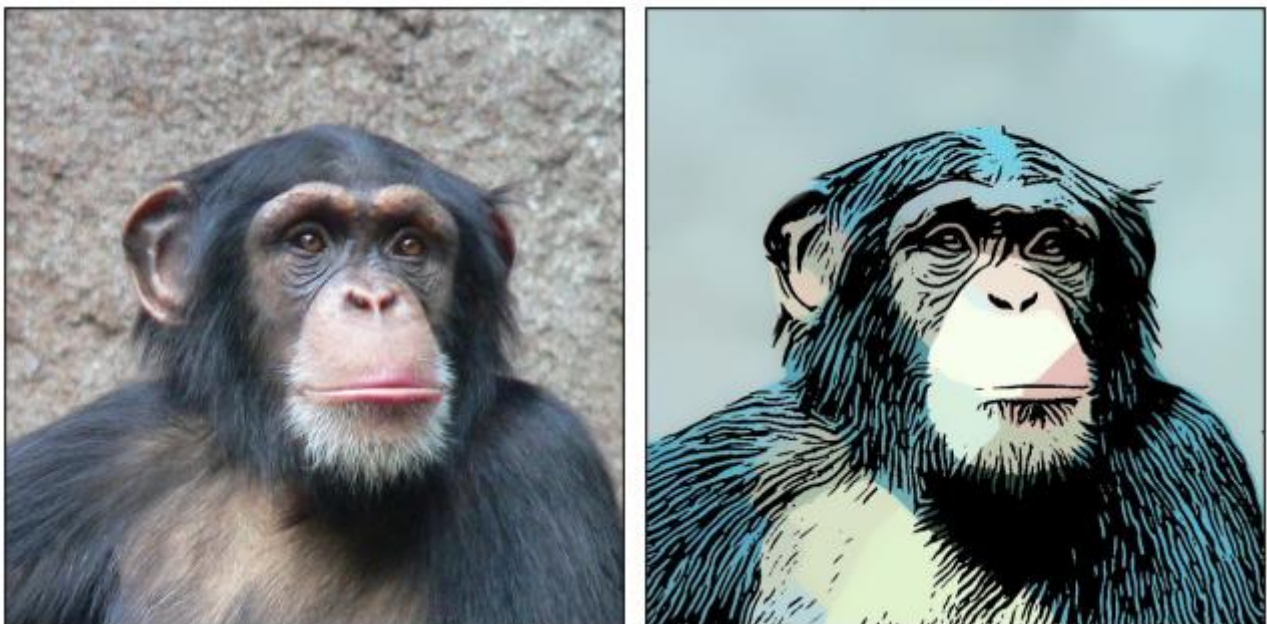


Fig.4.3. Conversion d'une photo de chimpanzé (à gauche) sous forme de dessin façon *Comics* (à droite), avec le



filtre "Engrave" de G'MIC.

## Triangulation de Delaunay.

Un algorithme de calcul de la [triangulation de Delaunay](#)<sup>W</sup> 2D/3D a été ajoutée à G'MIC (commande -*delaunay3d*), et le filtre *Artistic / Polygonize [delaunay]* du greffon *GIMP* l'utilise pour transformer une image sous la forme d'ensemble de triangles de *Delaunay*, qui s'adaptent plus ou moins aux contours de l'image. Chaque triangle généré peut prendre une couleur, soit aléatoire, soit constante, soit liée aux pixels de l'image qui appartiennent au triangle.

C'est un filtre permettant de créer des abstractions géométriques intéressantes.

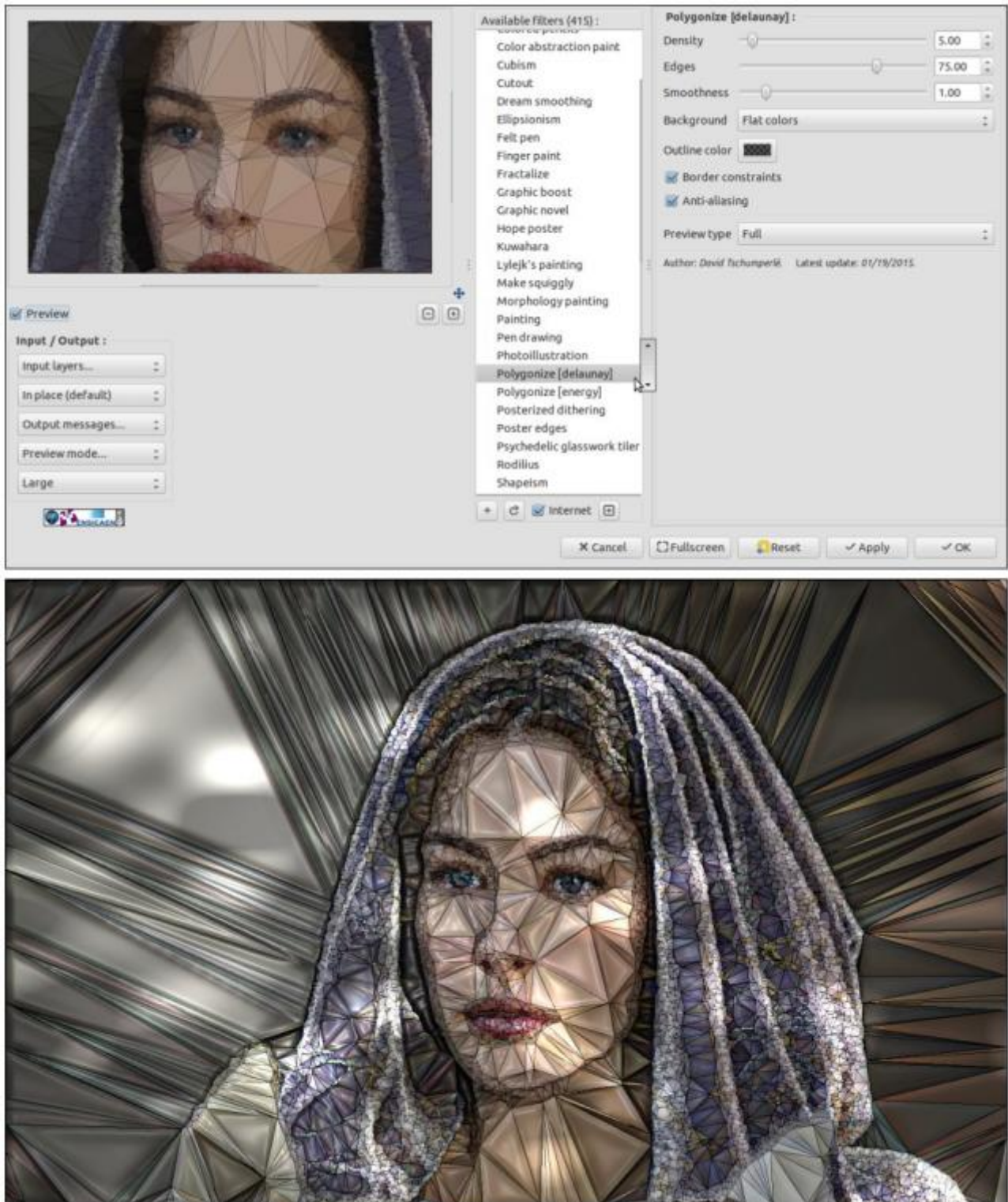


Fig. 4.4. Aperçu du filtre "Polygone [delaunay]" et application sur une image pour un rendu de type "vitrail".

C'est un filtre qui donne également des choses sympatiques sur des séquences d'images. Je me demande si l'importation de ces images triangulées sous [Blender](#) n'aurait pas un intérêt (et même plus loin, carrément un greffon *G'MIC* pour *Blender* ?).

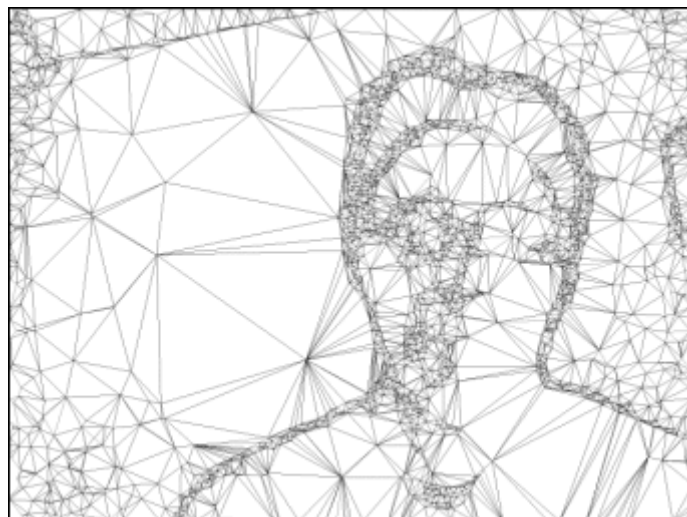


Fig.4.5. Application de la triangulation de Delaunay sur une séquence d'images.

### Autres filtres artistiques, en vrac.

Comme vous le voyez, *G'MIC* est un projet assez actif et le nombre de filtres disponibles (notamment artistiques) augmente de manière assez régulière. On ne peut donc pas décrire en détails tous les ajouts de ces dix derniers mois, et comme quelques images valent mieux qu'un long discours, voici un aperçu rapide de quelques-uns de ces filtres, présenté dans les figures ci-dessous.

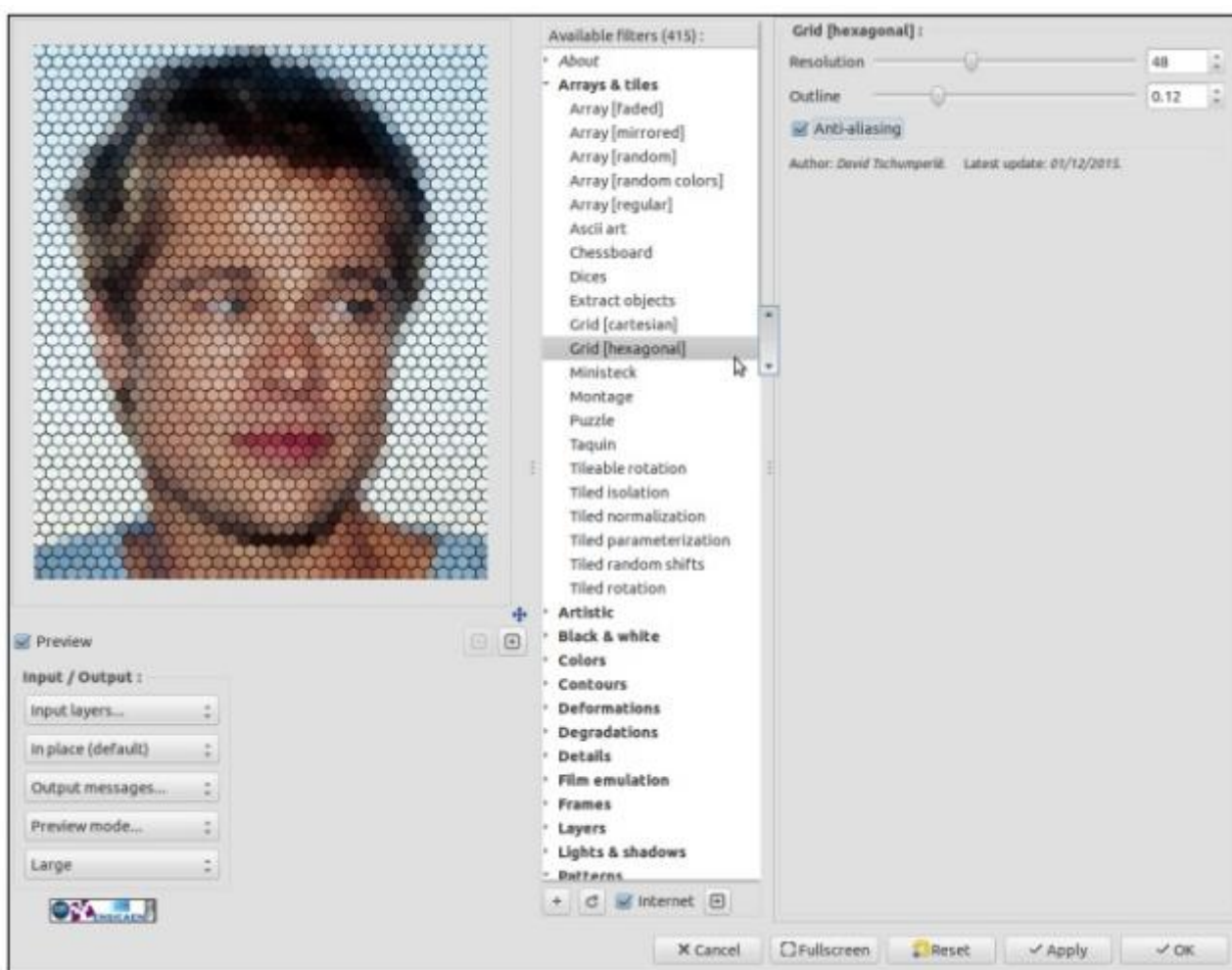


Fig.4.6. Aperçu du filtre "Arrays & Tiles / Grid [hexagonal]", potentiellement utile pour les créateurs de cartes des



jeux de type Wargames ?

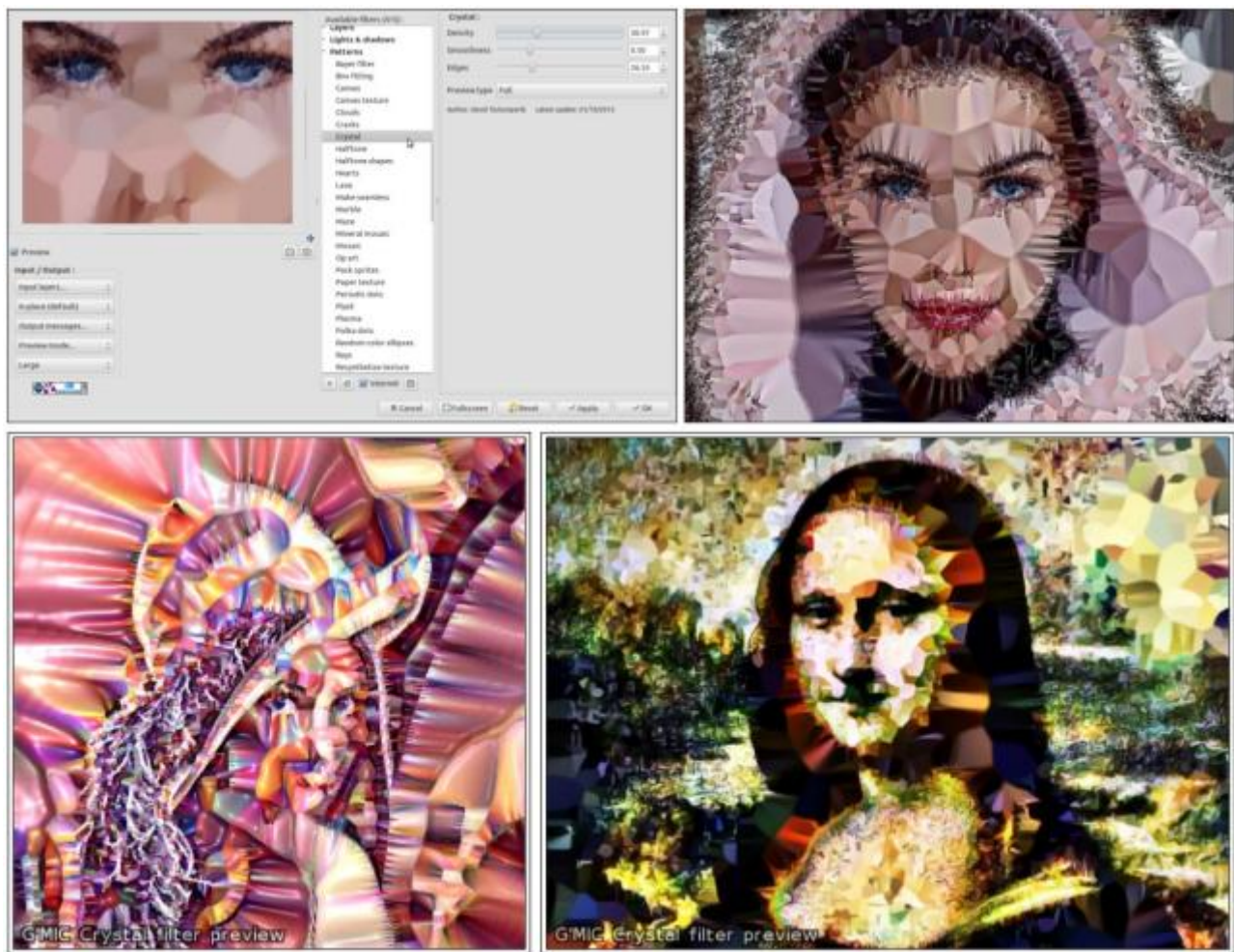


Fig.4.7. Aperçu du filtre "Patterns / Crystal" qui transforme vos images en cristaux multicolores.

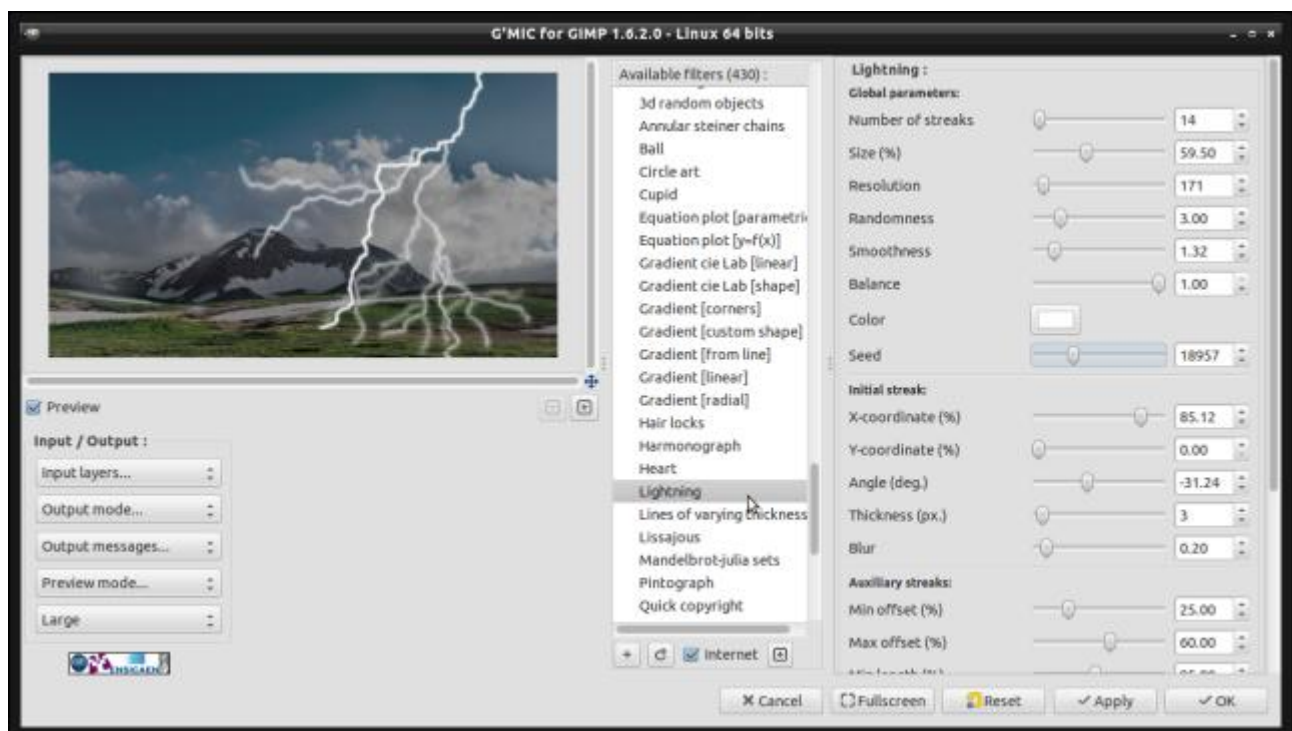


Fig.4.8. Aperçu du filtre "Rendering / Lightning" qui dessine un éclair dans une image.



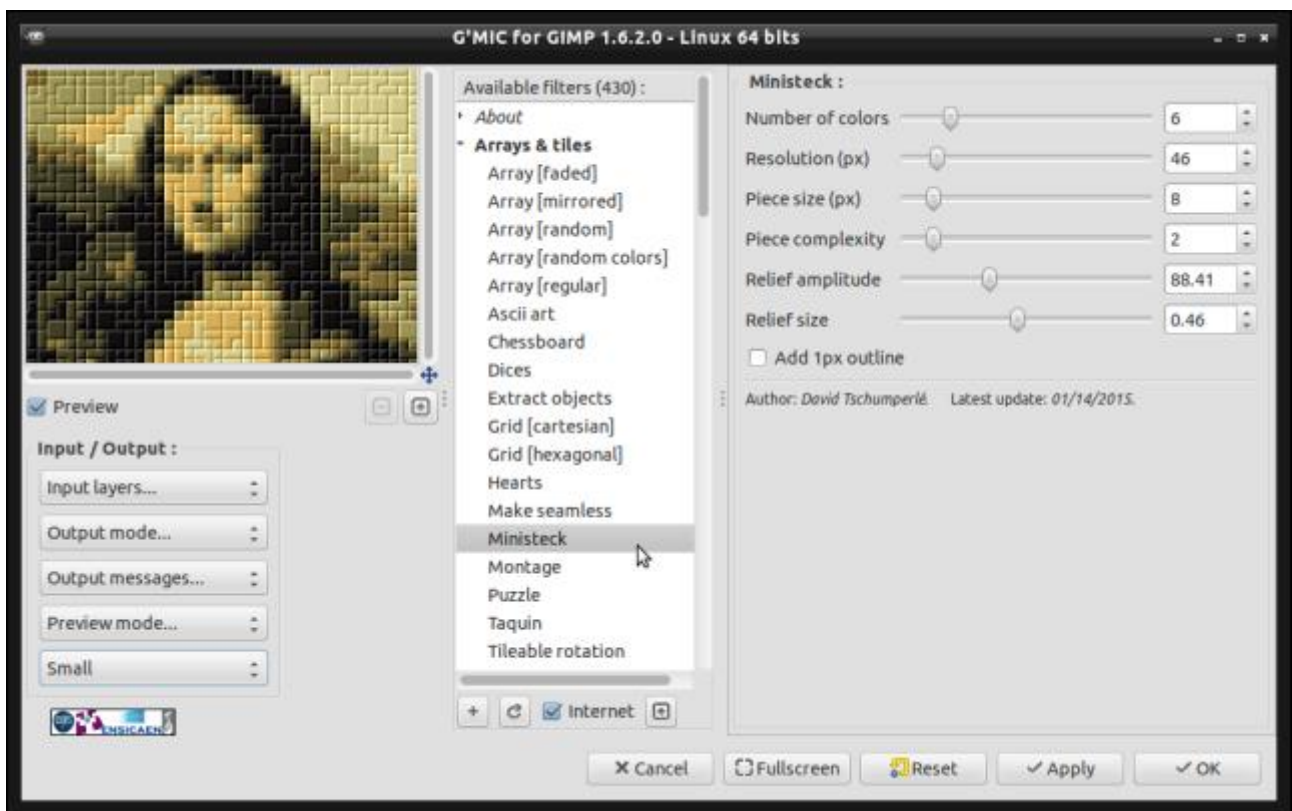


Fig.4.9. Aperçu du filtre "Arrays & Tiles / Ministeck" qui transforme vos images en représentation de type [Ministeck](#) (jeu pour enfants).

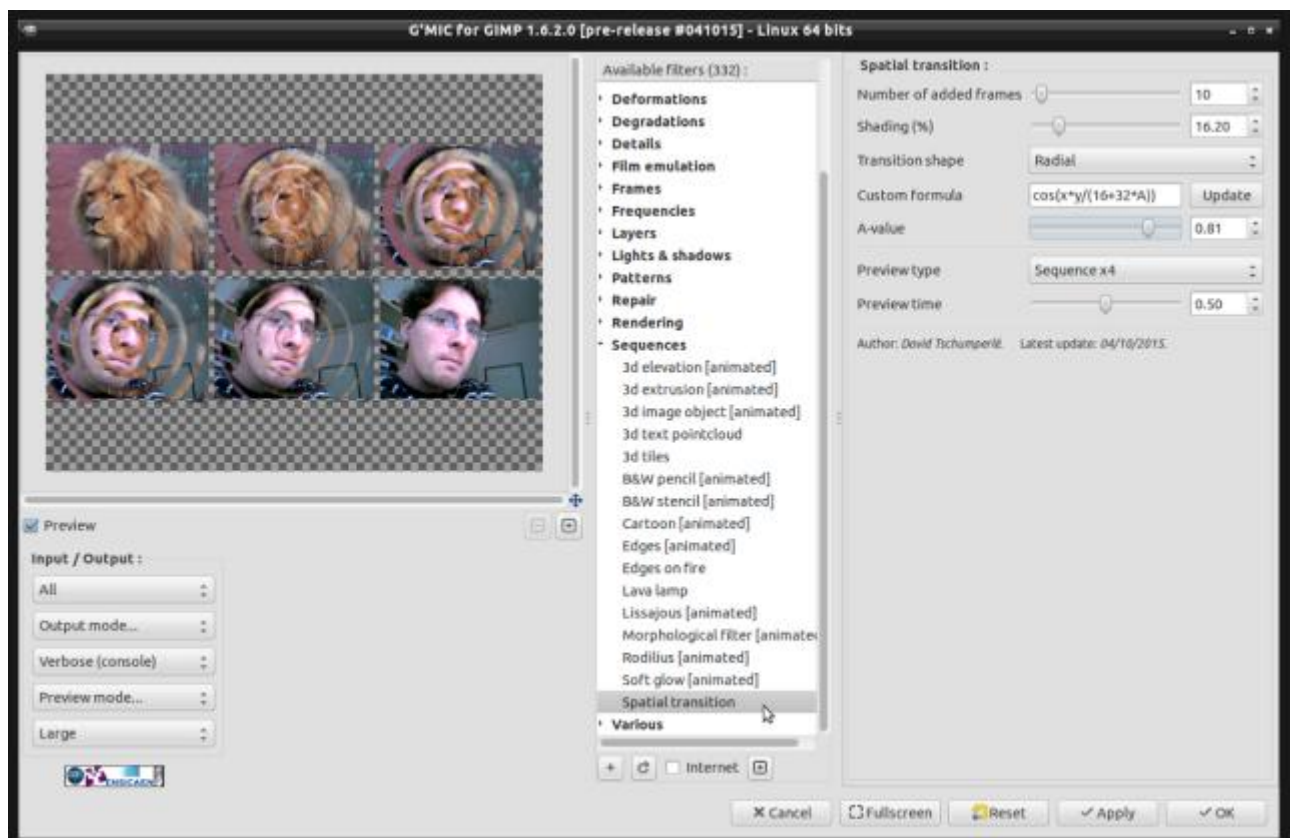


Fig.4.10. Aperçu du filtre "Sequences / Spatial transition", qui prend en entrée plusieurs calques et qui génère une séquence d'images de transition entre chaque calque consécutif, avec des motifs de transition réglables.

## 5. Autres faits notables.

Bien sûr, les quelques filtres présentés ci-dessus ne constituent qu'une toute petite partie (la plus visible) du travail effectué sur le code du projet G'MIC durant ces derniers mois. Voici en vrac, les autres aspects (plus techniques)

du logiciel qui ont été améliorés.

## Améliorations globales et bibliothèque *libgmic*.

- Énormément d'efforts ont été faits pour le nettoyage et l'optimisation du code : la taille de la bibliothèque *libgmic* a pu être réduite drastiquement (moins de 5 Mo actuellement), avec une API améliorée. L'utilisation de quelques fonctionnalités du C++11 (*rvalue-references*) permet d'éviter les copies intempestives de buffers temporaires, ce qui est vraiment appréciable dans le cas du traitement d'image où la mémoire allouée devient vite importante. Avec l'aide de [Lukas Tvrdy](#) développeur de *Krita*, nous avons pu améliorer la compilation du projet sous *Visual Studio* (Windows), appliquer des analyseurs statiques de code ([PVS Studio](#)), empiler des sessions de [valgrind](#) / [gprof](#)<sup>w</sup> / *g++* avec l'option `-fsanitize=address` pour obtenir un code plus propre et plus efficace. Pas vraiment drôle pour les développeurs, mais au final très satisfaisant pour l'utilisateur ! La conséquence directe de ce processus est de pouvoir proposer des versions *pre-release* intermédiaires qui restent stables tout en incorporant les dernières nouveautés ajoutées. La méthode de distribution de *G'MIC* se rapproche ainsi d'une méthode [rolling-release](#)<sup>w</sup>.
- La compilation du projet sous Windows a été également améliorée, avec l'utilisation de *g++-4.9.2 / MinGW* comme compilateur par défaut pour la génération des binaires, et la création d'installateurs pour les architectures 32 bits et 64 bits.
- *G'MIC* possède de nouvelles commandes pour compresser/décompresser des données quelconques à la volée (via l'utilisation de la [zlib](#)). De cette façon, nous avons pu réduire la taille des définitions de commandes, des exécutables, et accélérer le processus de mise à jour des filtres via Internet.
- Le projet *G'MIC* possède maintenant son propre nom de domaine <http://gmic.eu>, indépendant de *SourceForge*, son ancien hébergeur. Nous avons enrichi le contenu des pages web, grâce notamment à de très bons [tutoriels](#) sur l'utilisation de l'outil *gmic* en ligne de commande (écrits par [Garry Osgood](#), un grand merci à lui !). Une page à retenir pour tous ceux qui voudraient se mettre à *G'MIC* en douceur.

## Quelques nouveautés pour le filtrage d'images.

Pas mal de nouvelles commandes dédiées au filtrage d'image, et leurs entrées associées dans le greffon *G'MIC* pour *GIMP* ont été ajoutées : en vrac, des algorithmes de sharpening ("Mighty Details"), de déconvolution par noyau arbitraire (algorithme de [Richardson-Lucy](#)<sup>w</sup>), de [filtrage guidé](#) rapide, de [Non-Local-Means](#)<sup>w</sup> rapide, de régularisation type [Perona-Malik](#)<sup>w</sup>, de filtrage rapide à support fini, de [DCT](#)<sup>w</sup> (et son inverse) sont maintenant disponibles.

*G'MIC* possède aujourd'hui une foultitude d'opérateurs pour le filtrage linéaire et non-linéaire d'image, qui en fait un logiciel libre à posséder absolument pour les traiteurs d'image de tout poil (ne serait-ce pour les méthodes de débruitage qu'il propose). Pour les utilisateurs du greffon, cela signifie potentiellement des filtres encore plus riches dans le futur. Ci-dessous, un aperçu d'application du filtre *Details / Mighty details* sur une image de portrait, à partir du greffon pour *GIMP*, pour un effet de type [Dragan](#).

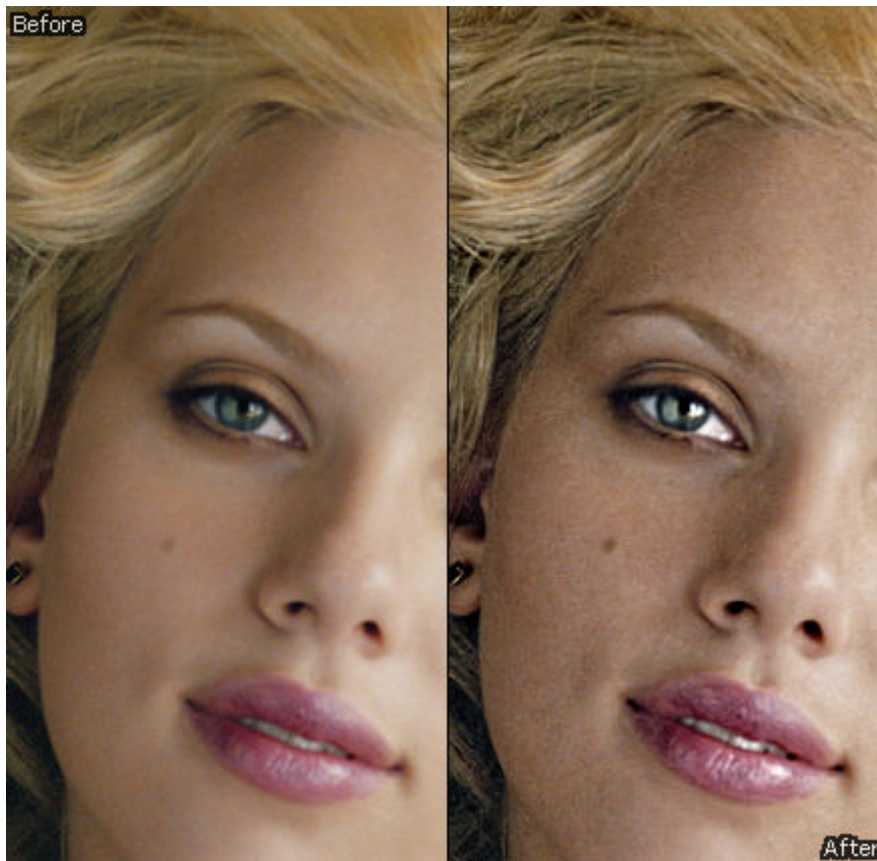


Fig.5.1 Application du filtre « Mighty Details » sur un portrait pour le rehaussement de détails.

### Amélioration du greffon G'MIC pour GIMP.

Le greffon G'MIC pour GIMP étant l'interface de G'MIC la plus utilisée à ce jour, il était important de lui apporter son lot d'améliorations :

- Le greffon possède maintenant un système de mise à jour automatique qui permet d'avoir une liste de filtres avec toutes les dernières améliorations/nouveautés. Au jour de la sortie de la version 1.6.2.0, le greffon possède 430 filtres différents à appliquer sur vos images, auxquels s'ajoutent 209 filtres qui sont considérés comme étant en phase de test (Catégorie *Testing*). Pour un greffon de 5,5 Mo, on peut considérer que ça fait un bon rapport *nombre de filtres / mémoire occupée* ! Une liste de l'ensemble des filtres disponibles dans le greffon est [visible ici](#).
- Lors de l'exécution d'un filtre un peu gourmand en temps de calcul, le greffon affiche maintenant quelques informations sur le temps écoulé et l'utilisation mémoire du filtre dans la barre de progression correspondante.

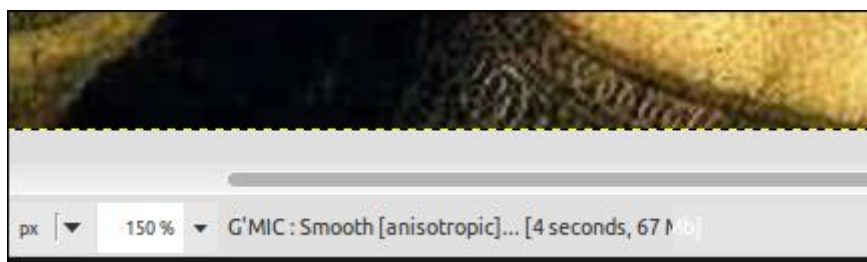


Fig.5.2. Affichage des ressources prises par un filtre dans le greffon G'MIC pour GIMP.

- L'interface du greffon a été améliorée : la fenêtre de pré-visualisation est plus précise dans le cas des filtres prenant plusieurs calques en entrées. Les filtres peuvent être maintenant interactifs : ils peuvent ouvrir leur propre fenêtre de visualisation et gérer des événements provenant de l'utilisateur. Un filtre peut



également décider de modifier l'interface de ses paramètres. Tout ceci nous a permis de proposer de nouveaux filtres interactifs intéressants, comme celui pour la colorisation de BDs, pour le réglage des courbes couleurs, ou encore pour l'extraction d'objets (décrits plus haut), qu'il n'était pas possible de réaliser dans le greffon pour GIMP précédemment (ça n'était possible qu'avec l'outil *gmic* en ligne de commande).

- Les filtres du greffon ont maintenant connaissance des informations de base des calques d'entrées (label, position, opacité, mode de mixage), et peuvent modifier ces informations sur les calques de sortie.
- Un nouveau filtre *About / User satisfaction survey* a été ajouté, permettant aux utilisateurs de donner quelques informations sur leur utilisation de G'MIC. Le résultat de ce sondage est [visible en temps réel ici](#). Bien sûr, cette image est elle-même générée par une commande G'MIC :).

## Débrider *G'MIC* : utilisation en ligne de commande.

[gmic](#) est l'interface « ligne de commande » de *G'MIC*, qui s'utilise à partir d'un shell. C'est l'interface la plus puissante de *G'MIC*, sans limitations inhérentes aux contraintes d'entrées-sorties des autres interfaces (par exemple, images 2D à 8bits/canal et 4 canaux maximum sous *GIMP*, une seule image d'entrée 2D avec *G'MIC Online*, etc.). Ici, on peut charger/sauver et manipuler des séquences d'images 3D multi-canaux à valeurs flottantes sans limites (autres que la mémoire disponible). L'interface ligne de commande a également connu son lot d'améliorations.

- [OpenCV](#) est maintenant utilisé par défaut pour charger/sauver des séquences d'images. On peut donc appliquer des traitements *frame* par *frame* sur des vidéos (commande *-apply\_video*), des streamings de la webcam (commande *-apply\_camera*), ou générer des fichiers vidéos à partir d'images fixes par exemple. De nouvelles commandes *-video2files* et *-files2video* pour décomposer des fichiers vidéos en plusieurs *frames* ou les recomposer ont été ajoutées. Le traitement de fichiers vidéos devient maintenant (presque) un jeu d'enfant avec *G'MIC* en ligne de commande ! L'aide (commande *-h*) a également été améliorée, avec une sortie en couleur, des corrections proposées en cas d'erreur de frappe sur une commande, des liens éventuels vers des tutoriels, etc.

```
~$ gmic -h blut
gmic: GREYC's Magic for Image Computing.
      Version 1.6.2.0, Copyright (c) 2008-2015, David Tschumperle.
      (http://gmic.eu)
[gmic] Command 'blut' has no description (did you mean '-blur' ?). Try 'gmic -h' for global help.
~$ gmic -h blur
gmic: GREYC's Magic for Image Computing.
      Version 1.6.2.0, Copyright (c) 2008-2015, David Tschumperle.
      (http://gmic.eu)

-blur (+):
          std_variation>=0[%], boundary, kernel |
          axes,std_variation>=0[%],_boundary,_kernel

Blur selected images by a quasi-gaussian or gaussian filter (recursive implementation).
(eq. to '-b').

'boundary' can be { 0=dirichlet | 1=neumann } and 'kernel' can be { 0=quasi-gaussian (faster) |
1=gaussian }.
When specified, argument 'axes' is a sequence of { x | y | z | c }.
Specifying one axis multiple times apply also the blur multiple times.

Default values: 'boundary=1' and 'kernel=0'.

Example: image.jpg --blur 5,0 --blur[0] 5,1
         image.jpg --blur y,10%

Tutorial: http://gmic.eu/tutorial/_blur.shtml
```

Fig.5.3. Aperçu du fonctionnement de l'aide en ligne de commande.

- L'appel à *gmic* sans arguments en ligne de commande active maintenant un mode *démo* où l'on peut sélectionner différentes petites animations interactives à partir d'un menu, pour voir ce que *G'MIC* a dans le ventre ! Une bonne occasion pour jouer à *Pacman*, *Tetris* ou au démineur en prétextant le fait de tester un logiciel de traitement d'images sérieux !

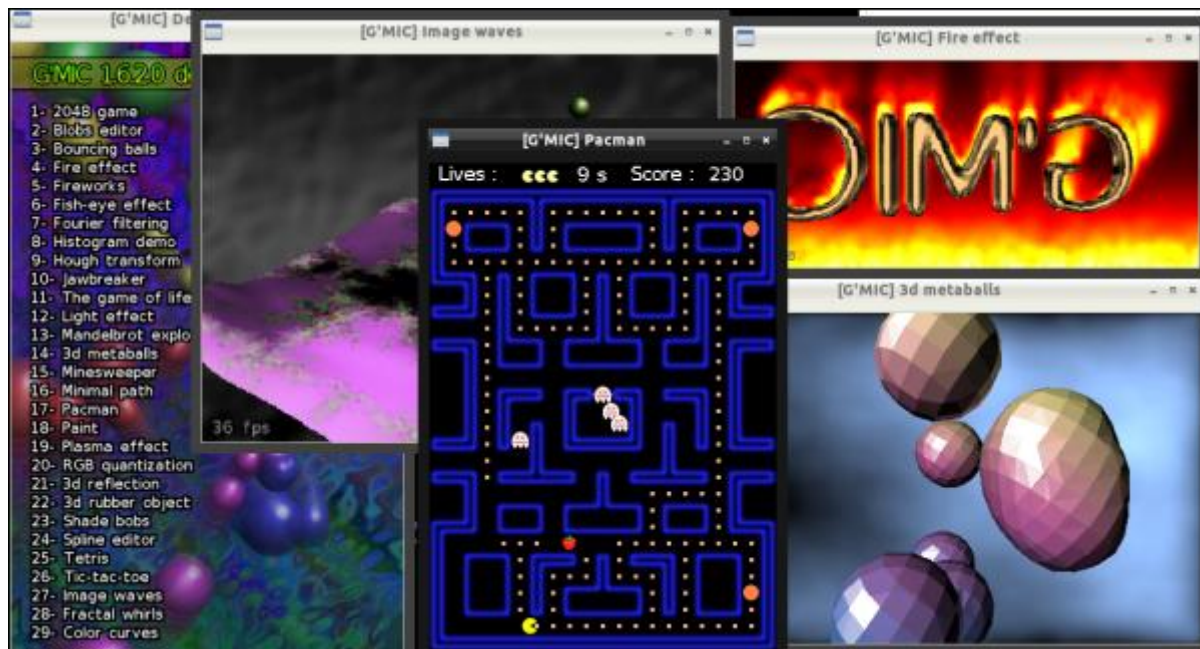


Fig.5.4. Aperçu de *G'MIC*, lancé en mode *démo* !

### Autres interfaces *G'MIC* en cours de développement.

- [ZArt](#) est une interface de *G'MIC* initialement développée comme plateforme de démonstration pour le traitement des images provenant d'une webcam. Nous l'utilisons classiquement lors de la Fête de la Science, pour illustrer ce que peuvent réaliser des algorithmes de traitement, pour le grand public. *ZArt* s'est enrichi significativement, d'une part en permettant d'importer des fichiers vidéos quelconques, et d'autre part, en proposant quasiment l'ensemble des filtres du greffon *G'MIC* pour *GIMP*. [Cette vidéo](#) illustre ces nouvelles possibilités, avec par exemple la simulation de films photo argentiques sur une vidéo en temps réel. Il pourrait vraiment être très intéressant de disposer d'un greffon *G'MIC* pour des logiciels d'effets vidéos, comme le tout nouveau [Natron](#) par exemple. On a déjà des contacts avec eux, donc on peut espérer que ça puisse se faire dans un futur plus ou moins proche. À noter que *ZArt* est également passé sans douleur de Qt 4 à Qt 5.

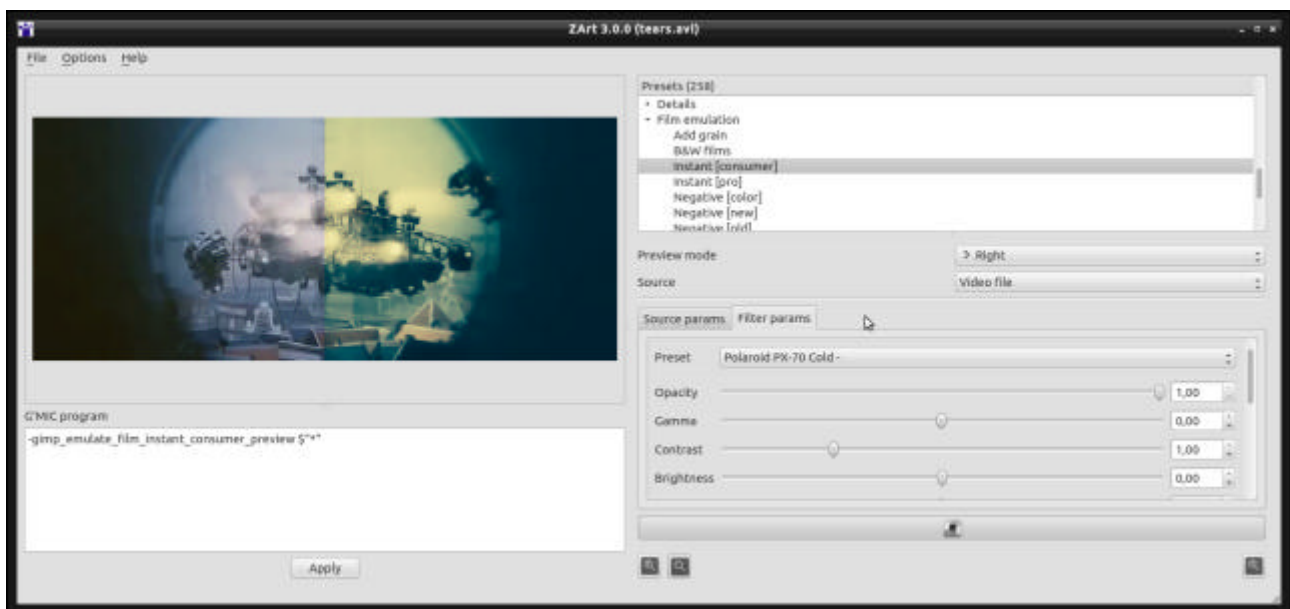


Fig.5.5. Aperçu du logiciel ZArt traitant une vidéo en temps réel.

- Depuis quelques temps, [Lukas Tvrđy](#) membre de l'équipe de développement du logiciel [Krita](#) a commencé à développer un greffon G'MIC pour Krita. Ce greffon est prometteur, il reprend l'ensemble des éléments du greffon déjà disponible pour GIMP, et il fonctionne déjà bien sous GNU/Linux. On travaille ensemble pour régler les quelques problèmes qui subsistent sous Windows. L'utilisation du greffon sous Krita est intéressante car il génère des résultats à 16 bits par composante, voire plus encore (flottants 32 bits), alors que le greffon pour GIMP est pour le moment limité à 8 bits par composante pour les entrées-sorties. C'est quelque chose qui pourrait changer dans le futur avec l'utilisation de GEGL, mais le manque de documentation actuel sur GEGL est un frein au support du 16 bits par composante dans le greffon G'MIC pour GIMP. Donc pour le moment, appliquer des traitements G'MIC sur des images en 16 bits par composantes ou plus nécessite l'utilisation de la ligne de commande ou du greffon Krita.

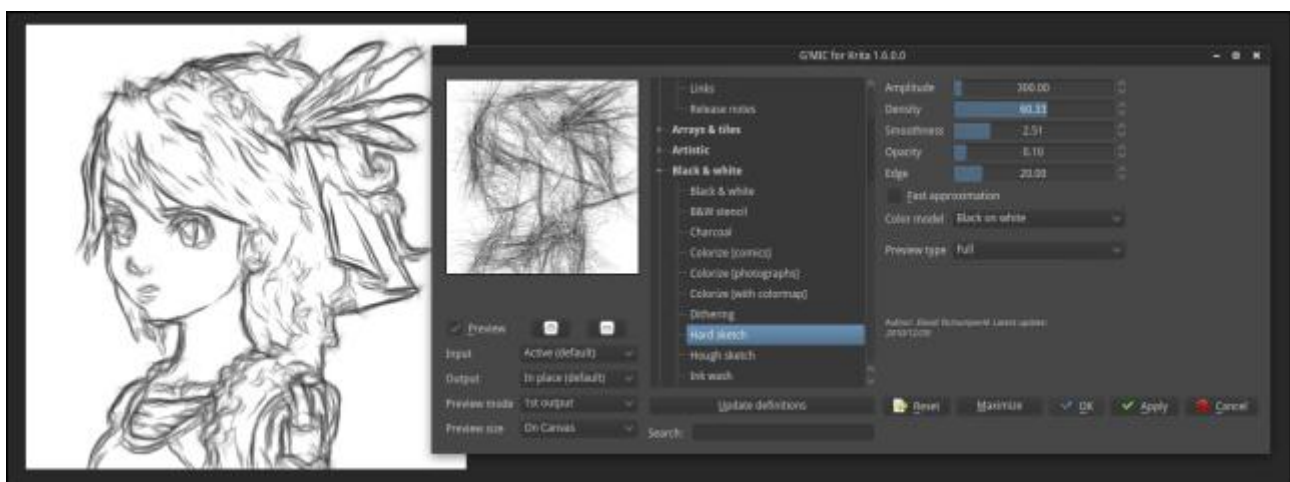


Fig.5.6. Aperçu du greffon G'MIC pour Krita en action.

(Cette image provient du site web de Krita : <https://krita.org/wp-content/uploads/2014/11/gmic-preview.png>)

- L'intégration de certains traitements de G'MIC dans [PhotoFlow](#) est également en cours. PhotoFlow est un logiciel libre très récent et prometteur qui se focalise sur le développement d'images RAW en JPEG, ainsi que sur la retouche photo.



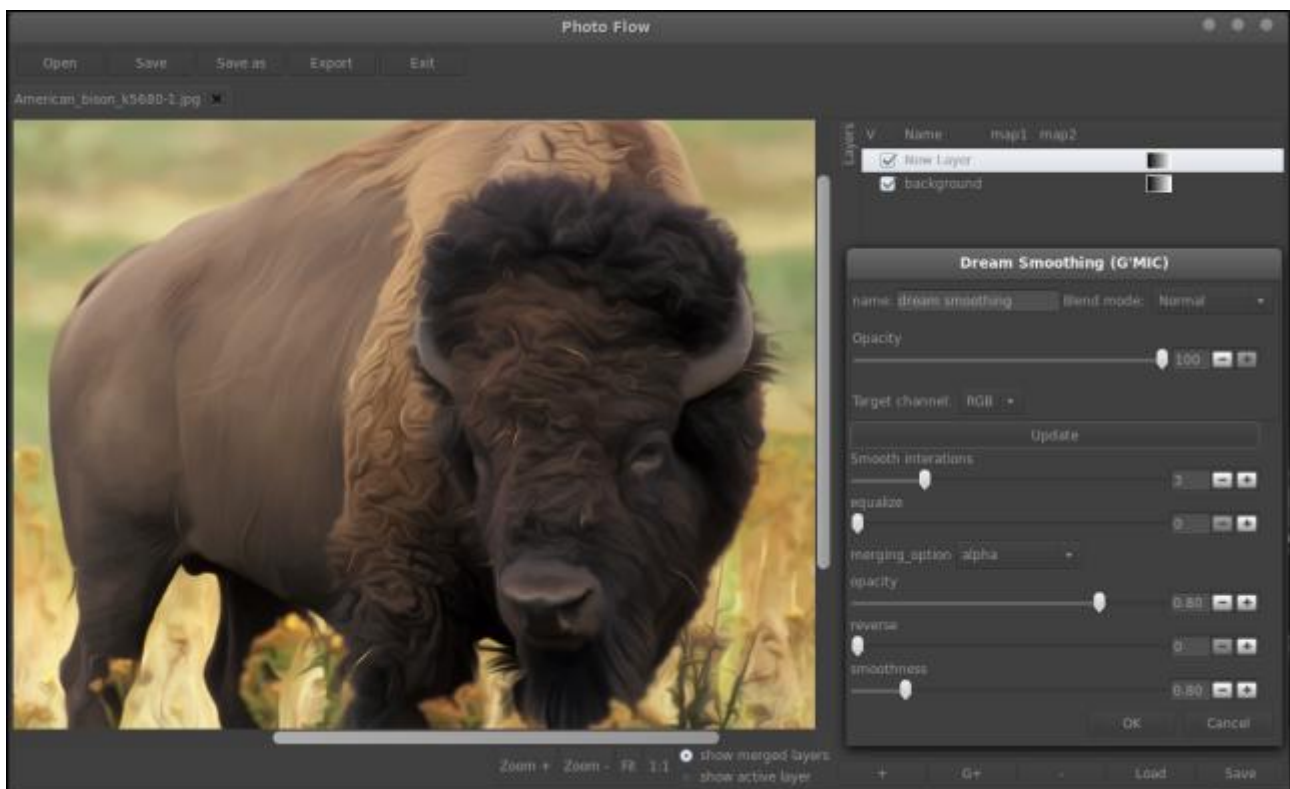


Fig.5.7. Aperçu du filtre "Dream Smoothing" de G'MIC inclus dans PhotoFlow.

(Cette image provient du site web de PhotoFlow : <http://photoflowblog.blogspot.fr/2014/12/news-gmic-dream-smoothing-filter-added.html>)

## 6. Perspectives et conclusions.




Voilà ! Le tour des nouveautés notables de ces dix derniers mois est terminé. À noter que tout ceci a été rendu possible grâce aux différents contributeurs (programmeurs, artistes, et utilisateurs en général) de plus en plus nombreux, donc merci à eux.

Dans le futur, on rêve de toujours plus d'intégration de la *libgmic* dans d'autres logiciels libres. Un greffon pour [Natron](#) serait idéal pour qu'on explore un peu plus les possibilités de traitement de vidéos avec G'MIC. Un greffon pour [Blender](#) serait pas mal aussi d'ailleurs ! Un noeud pour [GEGL](#) serait intéressant également, pour le prochain [GIMP 3.0](#) (mais bon, on a encore probablement une bonne dizaine d'années pour se décider avant que ça sorte :)).

Bref, on cherche toujours des contributeurs compétents et intéressés pour explorer ces différentes pistes (et les autres qu'on ne connaît pas encore), et on risque de pas s'ennuyer avec G'MIC dans les prochaines années.

Sur ce, j'y retourne, et je vous donne rendez-vous dans quelques mois pour des nouvelles fraîches du projet G'MIC !

## Aller plus loin

-  [Le projet G'MIC](#) (689 clics)
-  [Le greffon G'MIC pour GIMP](#) (428 clics)
-  [Documentation de 'gmic', l'outil en ligne de commande de G'MIC](#) (92 clics)
-  [Simulation de films argentiques avec G'MIC](#) (148 clics)
-  [Tutoriels illustrés sur l'utilisation de l'outil en ligne de commande](#) (130 clics)
-  [Changelog complet pour la version 1.6.2.0](#) (67 clics)

**Magnifique !**