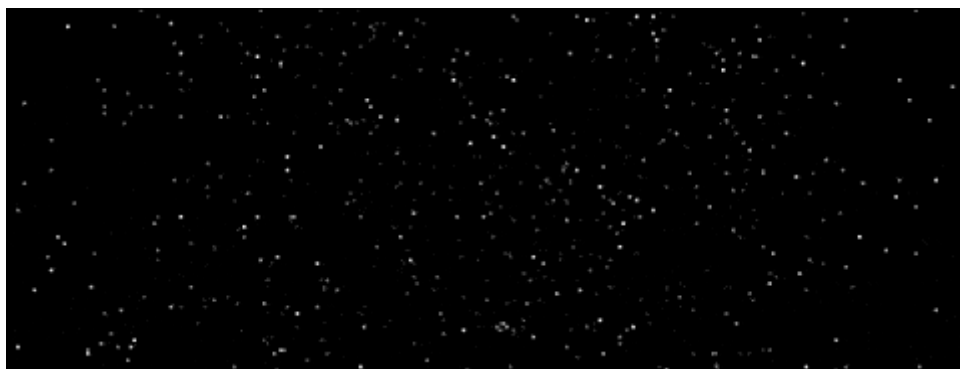


G'MIC 2.7.0 : Une rentrée pleine de style pour le traitement d'images !

Posté par [David Tschumperlé \(site web personnel\)](#) le 24/08/19 à 08:58. Édité par 5 contributeurs. Modéré par [Benoît Sibaud](#).
[Licence CC By-SA](#).

Étiquettes : [greyc](#) , [g'mic](#) , [traitement_d'images](#) + [Étiqueter](#)

L'équipe [IMAGE](#) du laboratoire [GREYC](#) est ravie de vous annoncer la sortie (pour la rentrée ☺), de la version **2.7.0** de [G'MIC](#) (*GREYC's Magic for Image Computing*), son [cadriciel](#)^w libre, générique, extensible, et sans doute un peu magique, pour [le traitement des images](#)^w.



[La dernière dépêche LinuxFr.org](#) sur ce logiciel libre avait été publiée, il y a un an, en août 2018. C'est donc l'occasion pour nous de résumer les nouveautés et les événements importants qui ont marqué la vie du projet le long de ces douze mois écoulés. Attachez vos ceintures, la route est longue et pleine de surprises !

Sommaire

- [1. G'MIC en 300 mots](#)
- [2. Donner du style à ses images](#)
- [3. Déformation interactive et morphing](#)
- [4. Toujours plus de transformations colorimétriques...](#)
- [5. Créer des palettes de couleurs par mélange de nuances](#)
- [6. Il y en a un peu plus,,j'veus le mets quand même ?](#)
- [7. Autres faits notables dans l'évolution du projet](#)
 - [7.1. Acceptation des dons](#)
 - [7.2. Intégration de la « Colorisation intelligente » dans GIMP](#)
 - [7.3. Brèves diverses liées au projet](#)
- [8. Le futur, et promis on arrête !](#)

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

1. G'MIC en 300 mots

[G'MIC](#) est un logiciel développé depuis plus de [10 ans](#) maintenant, principalement en [C++](#)^w, par deux membres de l'équipe [IMAGE](#) du [GREYC](#) : [Sébastien Fourey](#) et [David Tschumperlé](#). Il est distribué sous licence libre [CeCILL](#). Le GREYC est un laboratoire de recherche public français situé à Caen, spécialisé en sciences du numérique et chapeauté par trois tutelles : le [CNRS](#), l'[Université de Caen](#), et l'[ENSICAEN](#). L'équipe IMAGE, l'une des sept équipes du laboratoire, est composée de chercheurs, d'enseignants-chercheurs, de doctorants et d'ingénieurs, tous spécialisés dans les domaines de l'algorithmique et des mathématiques du [traitement d'images](#)^w.



Fig. 1.1 : logo du projet G'MIC, framework libre pour le traitement d'images, et sa mascotte « Gmicky » (réalisée par [David Revoy](#)).

G'MIC se veut multiplateforme (GNU/Linux, MacOS, Windows...). Il fournit un ensemble varié d'interfaces utilisateur pour la manipulation de données images *génériques*, à savoir des images ou des séquences d'images hyper-spectrales 2D ou 3D à valeurs flottantes (ce qui inclut de fait les images couleur « classiques »). Environ [un millier de fonctions](#) différentes de traitement sont déjà disponibles, nombre extensible à l'infini puisque les utilisateurs ont la possibilité de développer et d'ajouter leurs propres fonctionnalités via l'utilisation d'un langage de script intégré.

Les interfaces utilisateurs de G'MIC les plus utilisées sont : la commande [gmic](#), [exploitable en ligne de commande](#) (complément indispensable à [ImageMagick](#) ou [GraphicsMagick](#)), le service Web [G'MIC Online](#), mais surtout, le greffon [G'MIC-Qt](#), qui est accessible pour les logiciels grand public d'édition d'images [GIMP](#), [Krita](#) et [Paint.net](#), et qui permet de les enrichir de plus de 500 filtres et effets différents à appliquer sur des images.

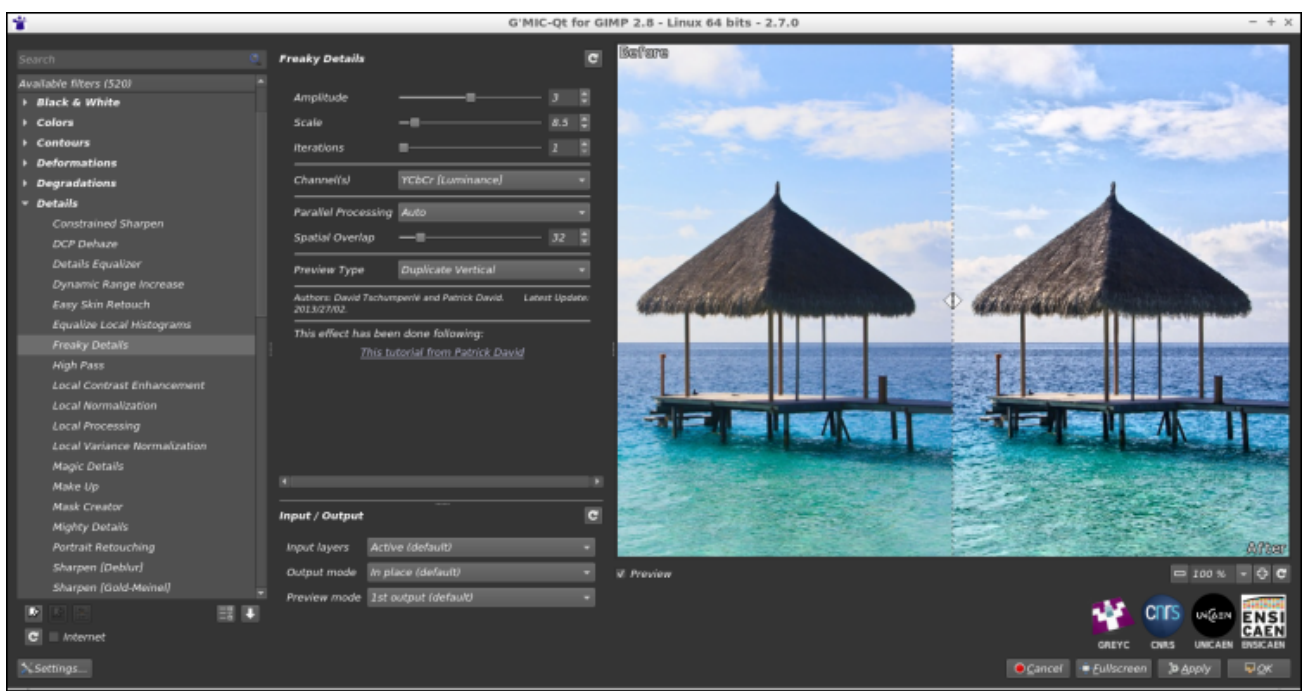


Fig. 1.2 : le greffon G'MIC-Qt, en version 2.7.0, aujourd'hui l'interface utilisateur du projet G'MIC la plus téléchargée.

Grâce à son architecture extensible, de nouveaux filtres de traitement d'images apparaissent régulièrement dans G'MIC, et c'est justement des derniers arrivés dont nous allons parler dans la suite de la dépêche.

2. Donner du style à ses images

G'MIC s'est doté récemment d'un filtre étonnant de **transfert de style** entre images, accessible depuis le greffon G'MIC-Qt, sous l'entrée « **Artistic / Stylize** ». Le concept du transfert de style est simple à appréhender : on

cherche à transformer une image (typiquement une *photographie*) en lui transférant le style d'une autre image (par exemple une *peinture*).



Fig. 2.1 : principe du transfert de style entre deux images.

L'implémentation effective d'un tel objectif est, quant à elle, relativement complexe. L'algorithme doit être capable de recomposer la photographie originale en « empruntant » des pixels à l'image de style et en les agençant intelligemment, tel un puzzle à reconstruire, pour coller au mieux au contenu des données à reproduire, en termes de contours, couleurs et textures. La facilité de la tâche dépend donc du niveau de compatibilité entre l'image et le style choisi. En traitement d'images, la plupart des implémentations existantes des méthodes de transfert de style se basent sur [des réseaux de neurones convolutionnels](#)^W, plus particulièrement des [réseaux antagonistes génératifs \(GAN\)](#)^W.

G'MIC implémente le transfert de style de manière différente (sans se baser sur des réseaux de neurones, l'article scientifique détaillant l'algorithme utilisé est en cours de rédaction!). Cette méthode est parallélisable et peut donc bénéficier de tous les cœurs de calculs présents sur la machine de l'utilisateur. Le temps de calcul dépend naturellement de la résolution de l'image à transformer, et de la précision de la reconstruction souhaitée. Sur un PC standard à 4 cœurs, comptez quelques dizaines de secondes pour des images à petite résolution (par ex. 800x800), à plusieurs minutes pour des images de tailles plus importantes.

Comme on peut l'imaginer, c'est un filtre **très polyvalent**, puisque l'on peut appliquer sans contraintes n'importe quel style sur n'importe quelle image d'entrée. Quelques peintures célèbres sont accessibles par défaut dans le filtre, afin de disposer de styles prédéfinis.

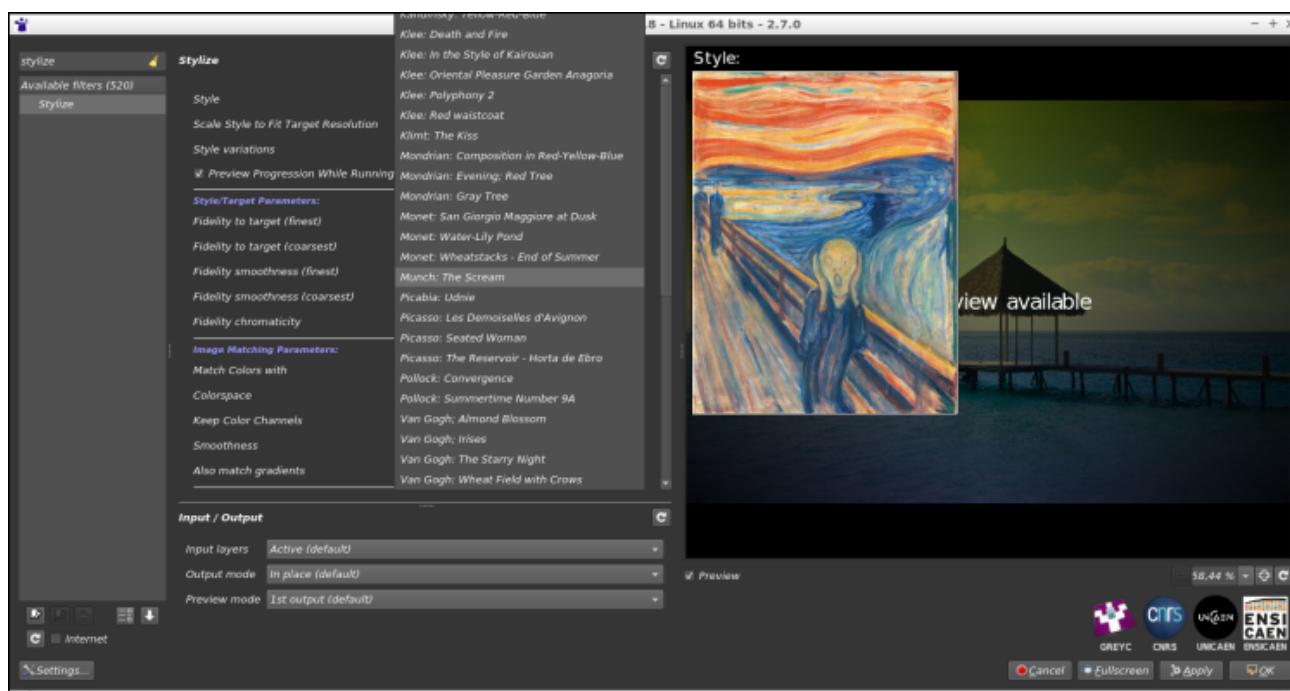


Fig. 2.2 : filtre « **Artistic/Stylize** », tel qu'il apparaît dans le greffon G'MIC-Qt, avec ses nombreux paramètres à régler!

Soyons honnêtes, il n'est pas toujours facile d'obtenir des résultats satisfaisants dès le premier jet. Il faut en général bien choisir ses images de départ, et jouer avec les nombreux paramètres disponibles pour affiner le type de

rendu généré par l'algorithme. Néanmoins, le filtre est capable de générer de belles images, comme celles illustrées ci-dessous (la photo originale est visible en haut à gauche, le style choisi en haut à droite, et le résultat du transfert de style en bas). Imaginez surtout le temps que cela prendrait à un graphiste de réaliser ces transformations « à la main » !

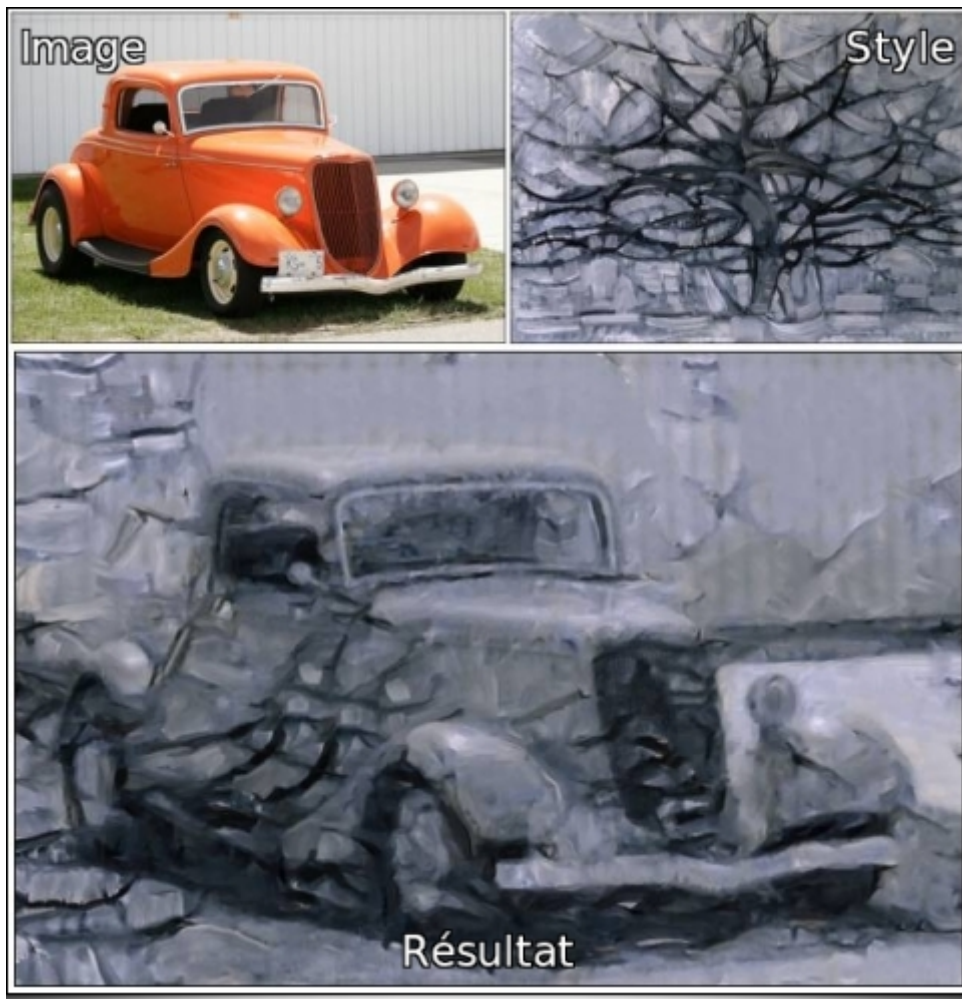


Fig. 2.3 : stylisation d'une photo de voiture à partir du tableau « [Arbre argenté](#)^W » de [Piet Mondrian](#)^W.



Fig. 2.4 : stylisation de la photo de voiture à partir du tableau « [Jaune-rouge-bleu](#) » de [Vassily Kandinsky](#).



Fig. 2.5 : stylisation de la photo de voiture à partir du tableau « [La Grande Vague de Kanagawa](#) » d'[Hokusai](#) »

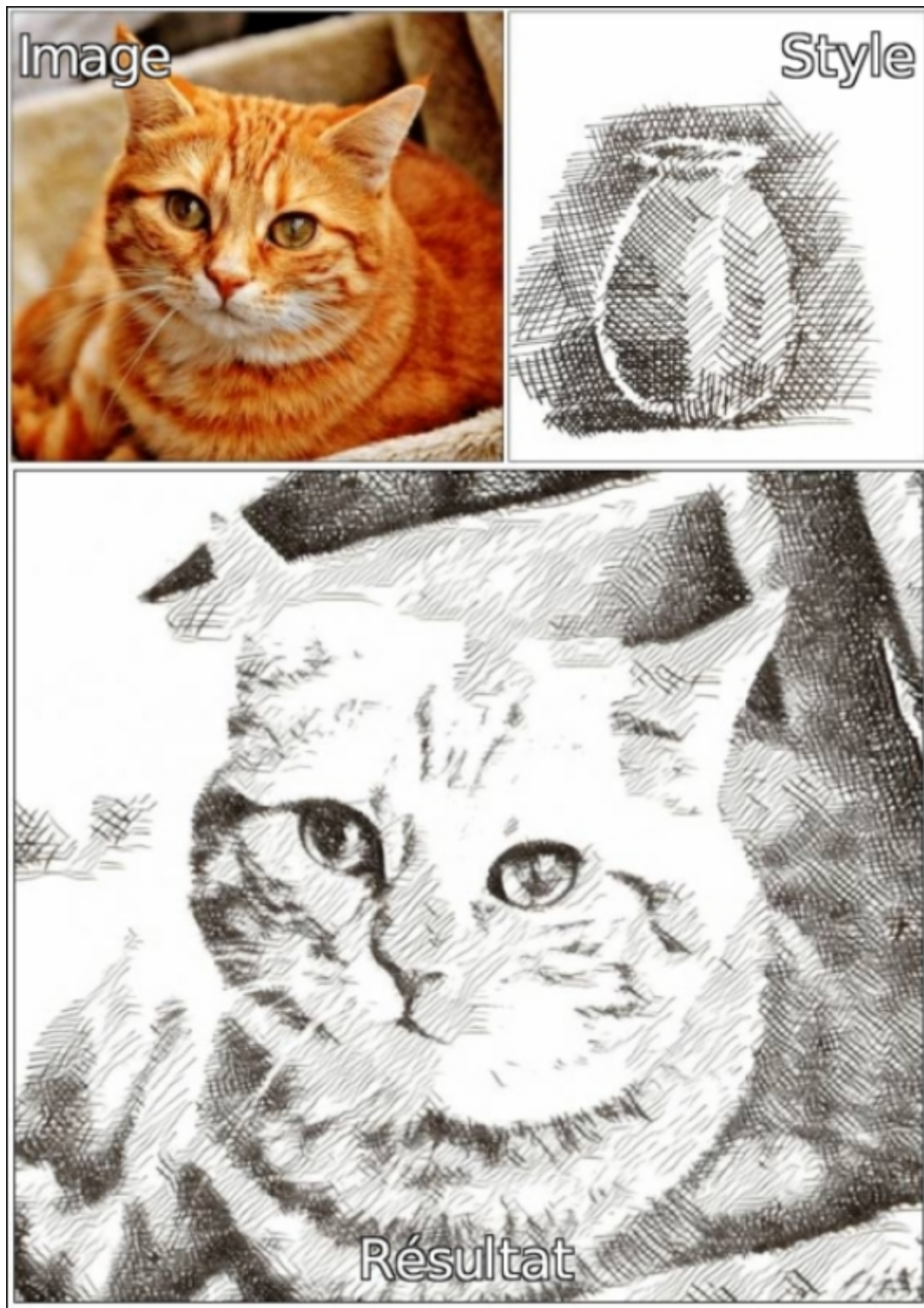


Fig. 2.6 : stylisation d'une photo de chat à partir d'un dessin hachuré.

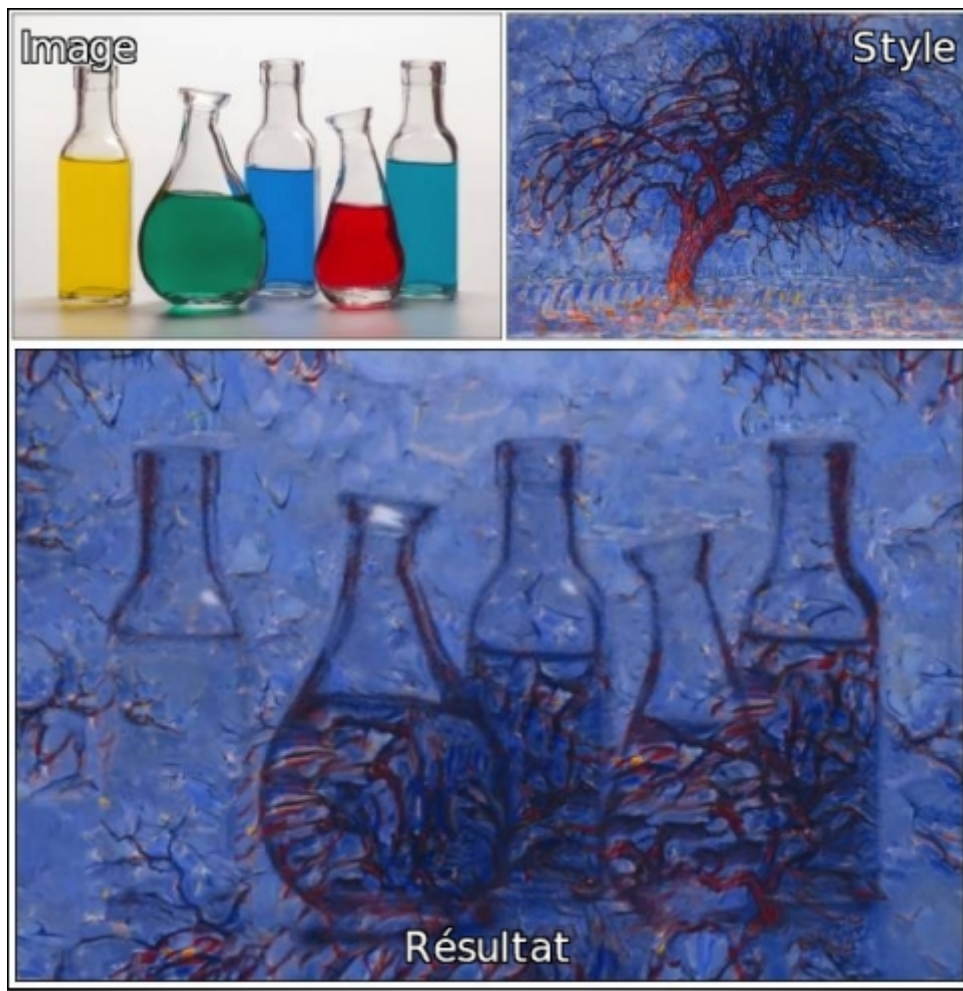


Fig. 2.7 : stylisation d'une photo de bouteilles à partir du tableau « [Arbre rouge^W](#) » de [Piet Mondrian^W](#).



Fig. 2.8 : stylisation de la photo de bouteilles à partir du tableau « [Le réservoir — Horta de Ebro](#) » de [Pablo Picasso](#)^W.

D'autres exemples de stylisation d'images sont visibles sur [la galerie d'images, dédiée à ce filtre](#). G'MIC est à notre connaissance, le seul logiciel de traitement d'images « grand public » proposant actuellement un filtre de transfert de style générique, où des images de style **quelconques** peuvent être choisies.

(*Second degré activé* : attendons encore une dizaine d'années, et [ça sera peut-être disponible dans Photoshop](#)!)

Dernière expérience amusante : récupérez une [photo de tête d'Alien](#), façon *Roswell*, et choisissez ensuite une image provenant de [l'ensemble de Mandelbrot](#)^W comme image de style. Utilisez le filtre de transfert pour générer une image façon « fractale » de votre tête d'Alien. Puis, faites croire au monde entier que l'ensemble de Mandelbrot contient la preuve mathématique de l'existence des extra-terrestres... ☺

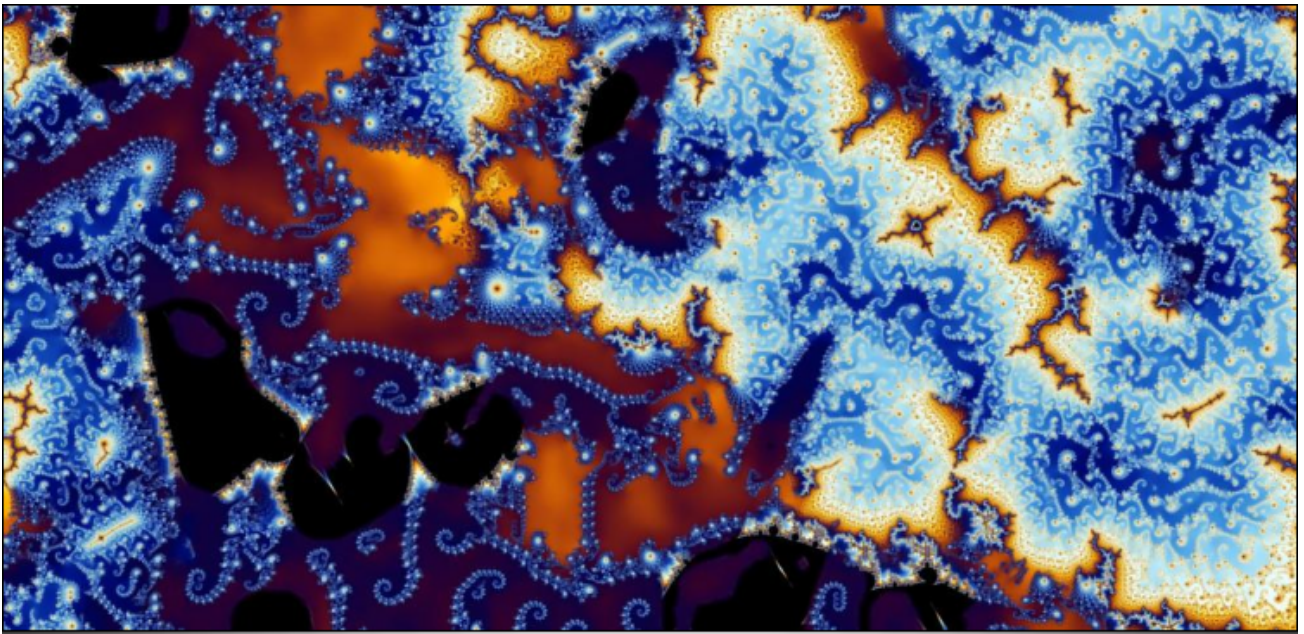


Fig. 2.9 : **Breaking News!** Une tête d'Alien a été trouvée dans l'ensemble fractal de Mandelbrot! (si vous ne l'apercevez pas du premier coup d'œil, penchez la tête vers la gauche...)

Bref, vous l'aurez compris, ce filtre a un certain potentiel créatif pour les artistes de tout poil !

3. Déformation interactive et morphing

Cette année, G'MIC s'est également pourvu d'une implémentation de [la méthode d'interpolation par RBF^w](#) (*Radial Basis Functions*), qui permet d'estimer une fonction dense interpolée en dimension quelconque, à partir d'un ensemble connu d'échantillons localisés de manière éparse (et pas forcément sur une grille régulière). C'est une invitation à développer des filtres dont l'interaction utilisateur se résume principalement à l'ajout, la suppression ou le déplacement de points-clés sur des images, le travail d'interpolation des données représentées par ces points-clés étant dorénavant géré automatiquement par G'MIC.

C'est donc ce que nous avons fait, avec deux nouveaux filtres bénéficiant de cette interpolation *RBF*.

Tout d'abord, le filtre « **Deformations/Warp [interactive]** » qui, comme son nom l'indique, propose à l'utilisateur de déformer localement une image en créant-déplaçant des point-clés.

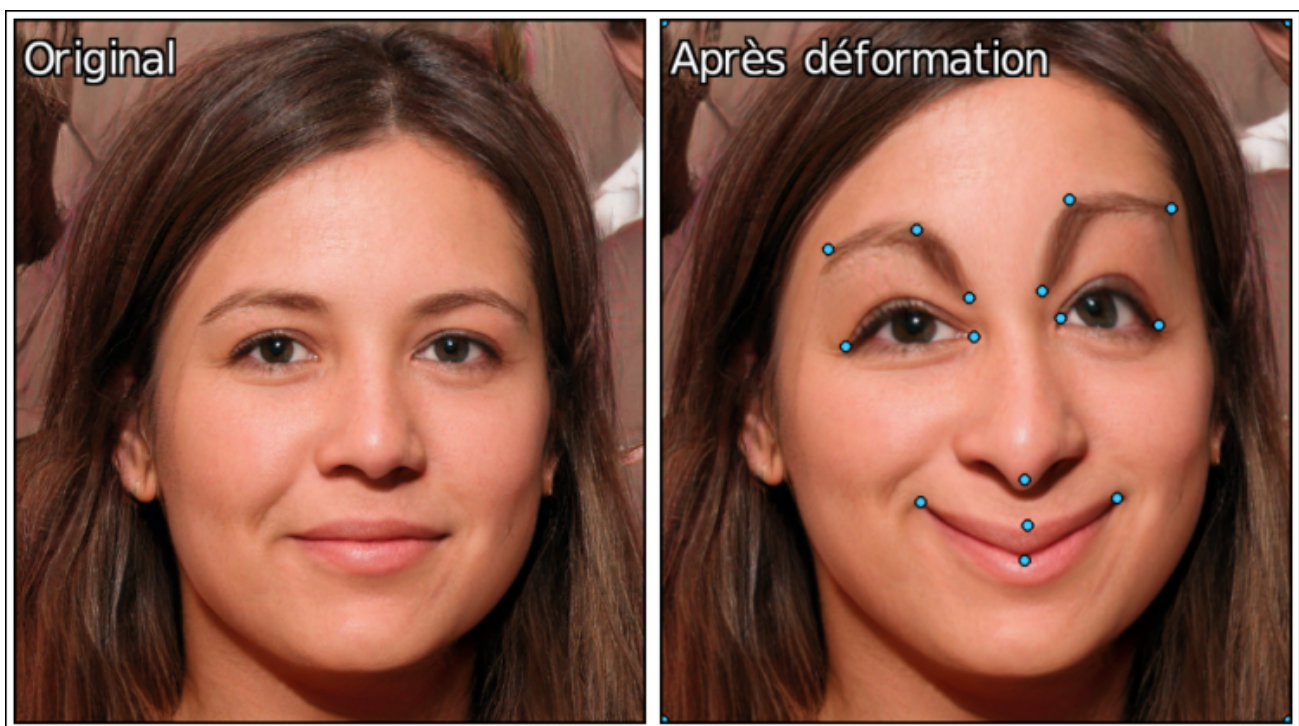


Fig. 3.1 : le nouveau filtre « Deformations/Warp [interactive] » permet de déformer interactivement des images, par exemple pour réaliser rapidement des caricatures à partir de photographies de portraits.

L'animation ci-dessous montre ce filtre interactif en cours d'utilisation, et illustre bien le comportement déformant de ces points-clés, qui peuvent être vus comme des points d'ancrage de l'image, qui sont déplacés de manière naturelle.



Fig. 3.2 : illustration de l'interaction utilisateur dans le filtre de déformation de G'MIC, basée sur la création et le déplacement de points-clés.

(À l'attention de celles et ceux qui pourraient s'émouvoir du sort que nous faisons subir aux personnes dont les photos sont utilisées dans les figures ci-dessus et ci-dessous : tous ces portraits sont totalement artificiels, aléatoirement générés par des GAN via le site [This Person Does Not Exist](https://thispersondoesnotexist.com/). Aucun préjudice moral en vue!).

Le grand avantage de l'interpolation *RBF* est de ne pas avoir à gérer explicitement une *structuration spatiale* entre les points-clés, par exemple par la définition d'un [maillage](#)^W (c'est-à-dire d'une « grille de déformation »). On obtient ainsi un plus grand degré de liberté dans les déformations possibles (voir fig. 3.3. ci-dessous). Et dans le même temps, on conserve un contrôle assez fin sur l'amplitude locale de déformation, puisqu'en ajoutant plus de points-clés « identités » autour d'une zone, on restreint naturellement l'amplitude de la déformation à l'intérieur de cette zone.



Fig. 3.3 : l'interpolation RBF permet de créer des déformations continues complexes, avec très peu de points-clés (ici, par inversion des positions des yeux droit/gauche, et seulement quatre points-clés utilisés).

Enfin, notons qu'une courte démonstration de ce filtre de déformation est visible dans [cette vidéo Youtube](#).

Et pourquoi ne pas étendre le principe de cette interpolation à deux images, au lieu d'une seule ? C'est justement ce que propose un autre nouveau filtre « **Deformations/Morph [interactive]** » qui réalise un [morphing^w](#) entre deux images (situées sur deux calques séparés), avec la même technique d'interpolation, qui demande à l'utilisateur de placer uniquement des points-clés colorés en correspondance sur les deux images.

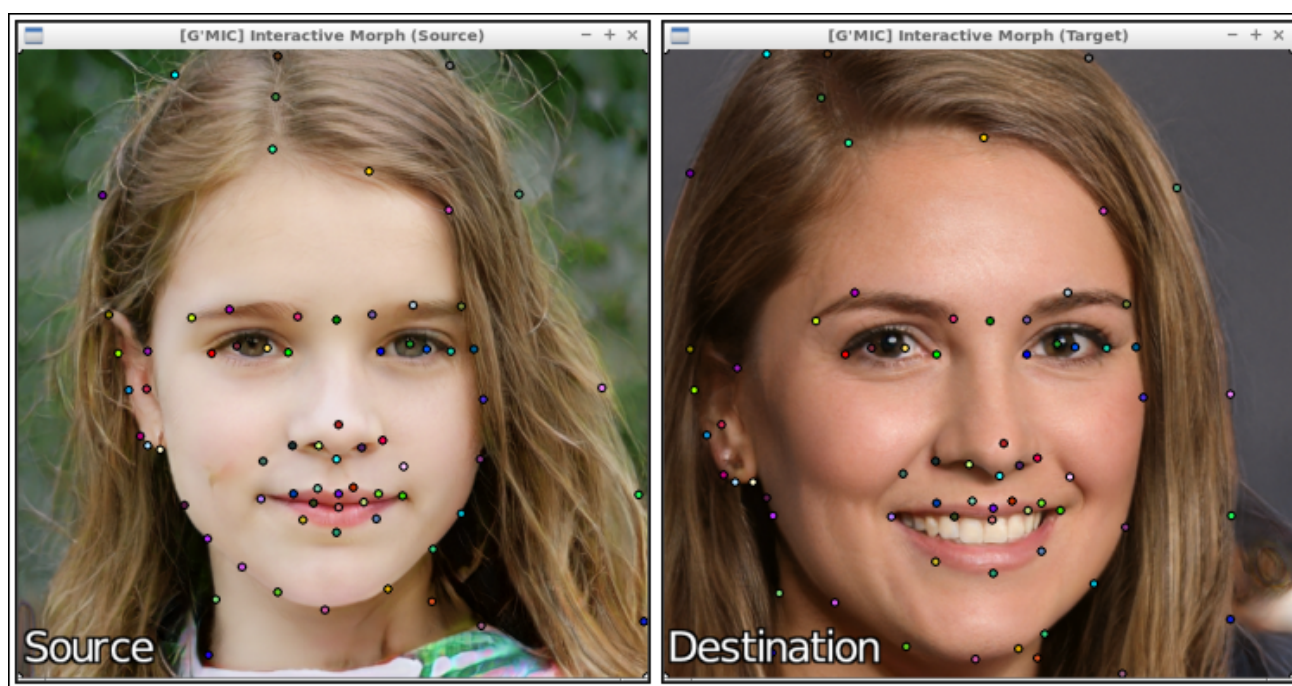


Fig. 3.4 : le filtre « Deformations/Morph [interactive] » demande à l'utilisateur de positionner des points-clés indiquant des zones de correspondance entre deux images.

Sur l'exemple ci-dessus, des points-clés sont placés sur des endroits caractéristiques des deux visages (bout du nez, des lèvres, des sourcils, etc.). En pratique, cela ne prend pas plus de cinq minutes. Grâce à ces correspondances, l'algorithme en déduit une carte de déformation globale d'une image vers l'autre, et peut ainsi générer des images « mixées » où les éléments du visage restent relativement bien alignés.

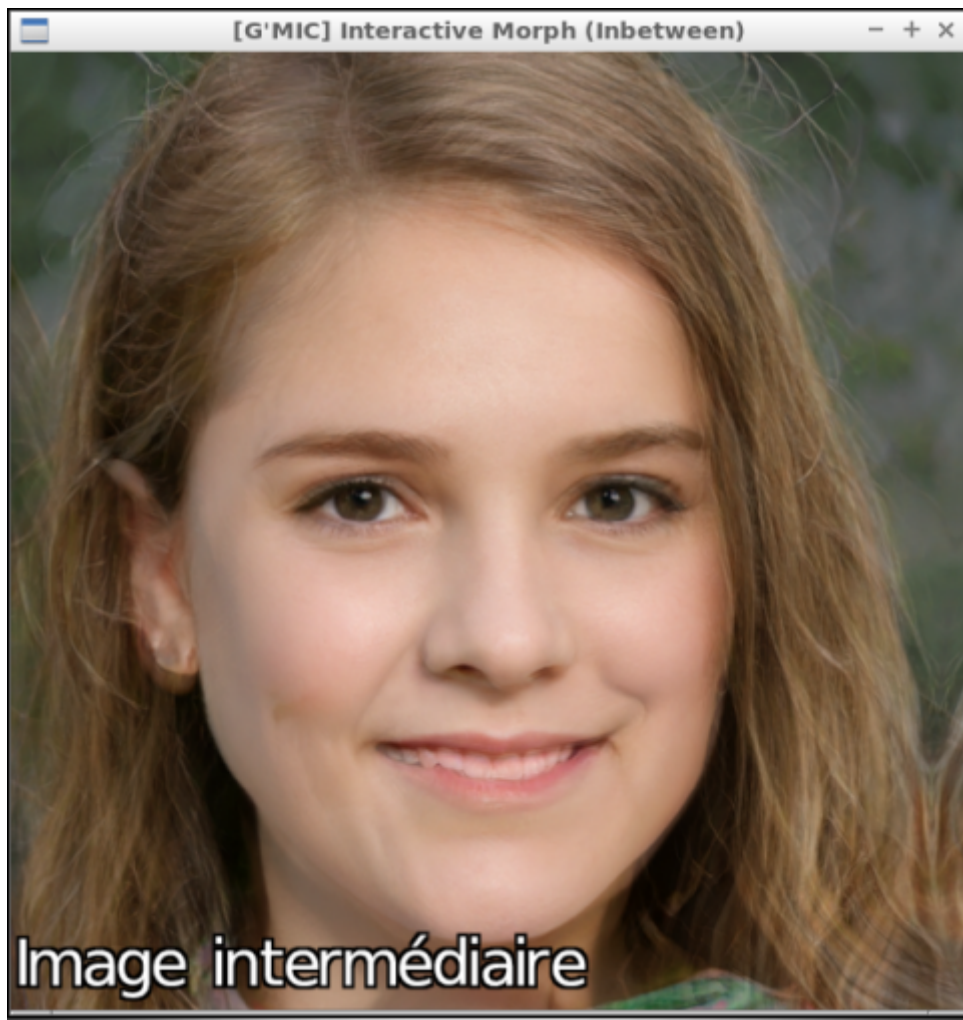


Fig. 3.5 : une des images intermédiaire générée pour le morphing entre les deux visages.

Par comparaison, voici ce qu'on obtiendrait en mélangeant simplement les deux images d'entrée entre elles, c'est-à-dire sans corriger la différence de localisation des éléments du visage entre les deux images. C'est pas joli joli !



Fig. 3.6 : un simple moyennage des images « Source » et « Destination » révèle les différences de localisation des éléments du visage.

Ainsi, le filtre de morphing est capable de générer rapidement un ensemble d'images intermédiaires, allant du visage « Source » au visage « Destination », séquence qui par la suite peut être sauvegardée sous forme

d'animation.



Fig. 3.7 : animation obtenue après génération de l'ensemble des frames intermédiaires par le filtre de morphing de G'MIC.

Beaucoup d'autres cas d'utilisation de ce filtre de morphing sont envisageables. L'exemple suivant illustre son application pour synthétiser une animation à partir de deux photographies du même objet (un nain de jardin), mais prises avec des profondeurs de champ^w différentes.



Fig. 3.8 : deux photographies à profondeurs de champ différentes, avec le placement des points de correspondance réalisé par l'utilisateur.



Fig. 3.9 : animation obtenue après génération de l'ensemble des trames intermédiaires par le filtre de morphing de G'MIC.

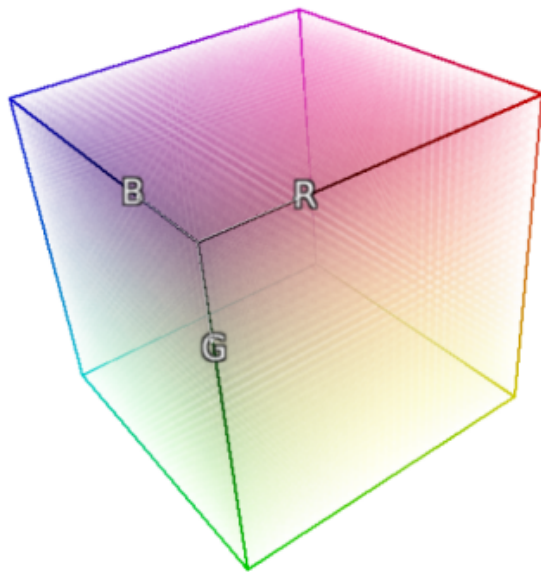
Les utilisateurs et les utilisatrices de la ligne de commande seront potentiellement intéressés d'apprendre que ces deux filtres peuvent être testés très simplement à partir d'un *shell*, de la façon suivante :

```
$ gmic image.jpg x_warp  
$ gmic source.jpg target.jpg x_morph
```

4. Toujours plus de transformations colorimétriques...

Depuis plusieurs années, G'MIC possède des filtres de transformations colorimétriques permettant, par exemple, de simuler le développement de photographies avec des pellicules argentiques, ou encore de donner des ambiances colorimétriques particulières à des images (ensoleillement, pluie, brouillard, lumière du matin, de l'après-midi, du soir, nuit, etc). Dans une [précédente dépêche](#), nous avons déjà évoqué ces filtres, qui se basent essentiellement sur l'utilisation de [CLUT 3D[™]](#) (*Color Lookup Tables*) pour la modélisation des transformations couleurs.

Une *CLUT* est tout simplement un tableau 3D qui fournit pour chaque couleur *RGB* existante, une couleur de remplacement à appliquer sur l'image.



(a) *CLUT* $F : RGB \rightarrow RGB$,
visualisée en 3D



(b) Image couleur originale



(c) Image après transformation par F

Fig. 4.1 : modélisation d'une transformation colorimétrique par une « Color LUT 3D ».

L'intérêt principal de ces *CLUT*, c'est la grande variété de transformations qu'elles permettent de représenter : elles peuvent en effet définir des fonctions de modifications de couleurs $RGB \rightarrow RGB$ avec des variations absolument quelconques. La seule « contrainte » de ces méthodes, c'est que tous les pixels d'une image ayant la même couleur seront transformés en des pixels également de couleurs identiques.



Image originale



"60's"



"Color Negative"



"Orange Tone"



"Ilford Delta 3200"



"Backlight Filter"



"Bleach Bypass"



"Late Sunset"

Fig. 4.2 : illustration de la variété des transformations colorimétriques modélisables par des *CLUT*.

L'inconvénient, par contre, c'est que ces *CLUT* 3D sont des données relativement *volumineuses*. Quand on veut en proposer plusieurs centaines différentes dans un même logiciel (ce qui est le cas dans *G'MIC*), on se retrouve vite avec un volume de données important à installer. Pour illustrer ces propos, prenons l'exemple d'un autre logiciel libre, [RawTherapee](#), qui propose sur son site de télécharger un pack additionnel de **294** fonctions *CLUT*, toutes stockées sous forme de fichiers `.png` dans une archive `.zip` d'une taille totale de **402 Mio**. Même si on peut considérer que stocker quelques centaines de [mébioctets](#)^W n'est plus forcément bloquant de nos jours, ça reste une taille pénalisante pour une distribution rapide et légère de filtres colorimétriques basiques.

Cette année, nous avons donc effectué au GREYC, un travail de recherche et de développement important autour de cette problématique. Le résultat : un nouvel algorithme de compression (*avec pertes imperceptibles*) capable de générer des représentations binaires de *CLUT* avec des taux de compression moyen **de plus de 99 %**, relativement aux données *déjà compressées sans pertes*. L'idée générale est de déterminer un ensemble de points-clés couleurs optimal qui permette la reconstruction d'une *CLUT* (*décompression*) avec une erreur de reconstruction minimale. Stocker ces points-clés est alors incontestablement plus léger que de stocker l'ensemble des couleurs transformées.

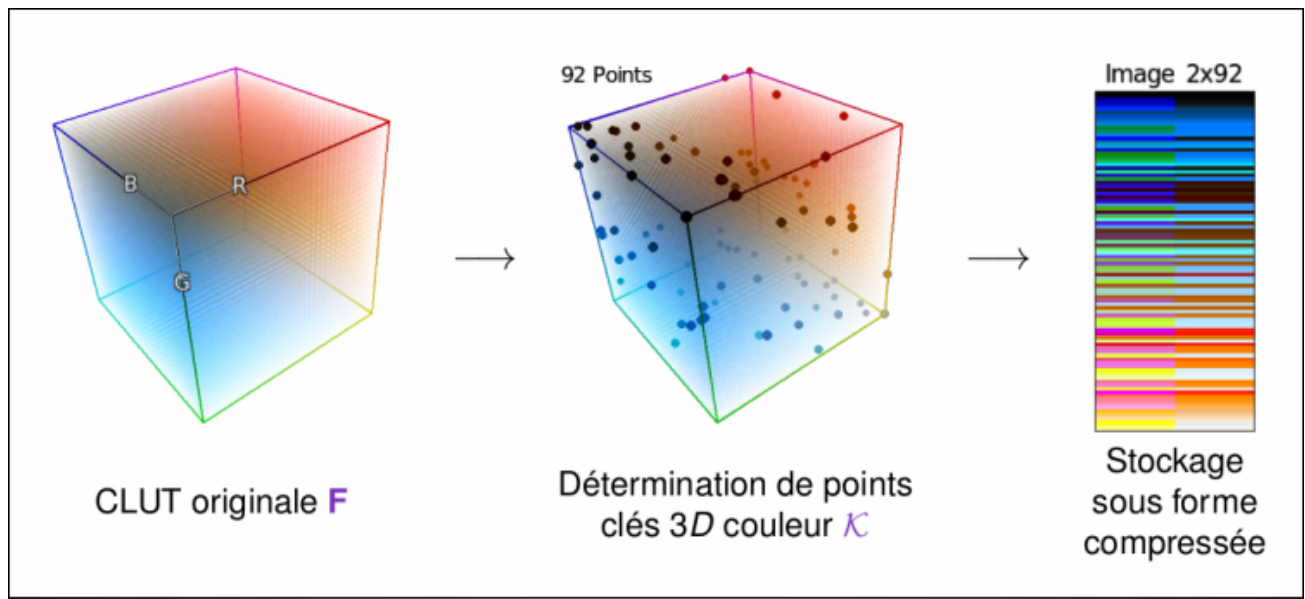


Fig. 4.3 : principe de la compression de *CLUT*, par stockage d'un ensemble de points-clés bien choisis.

Actuellement, l'exploitation de cette méthode de compression originale nous permet de mettre à disposition pas moins de **763** *CLUT* dans le logiciel *G'MIC*, toutes stockées dans un fichier binaire de **moins de 3 Mio** !

Tous ces filtres de variations de couleurs ont été regroupés dans deux entrées distinctes du greffon *G'MIC-Qt*, à savoir « **Colors/Simulate Film** » (pour toutes les simulations de films argentiques), et « **Colors/Color Presets** » (pour toutes les autres transformations). Chacun de ces filtres propose des sous-catégories pour structurer l'accès aux différentes *CLUT*. À notre connaissance, cela fait de *G'MIC*, le logiciel de traitement d'images détenteur du record mondial du nombre de transformations couleurs proposées, et ce, en gardant une taille raisonnable.

Le lecteur, ou la lectrice, intéressé par les détails mathématiques de ces algorithmes de compression/décompression de *CLUT*, pourra consulter [l'article](#) que nous avons publié, ainsi que les [transparents](#) des présentations qui seront faites dans les prochains jours aux conférences sur le traitement d'images [GRETSI'2019](#) (conférence nationale, à Lille) et [CAIP'2019](#) (conférence internationale, à Salerne).

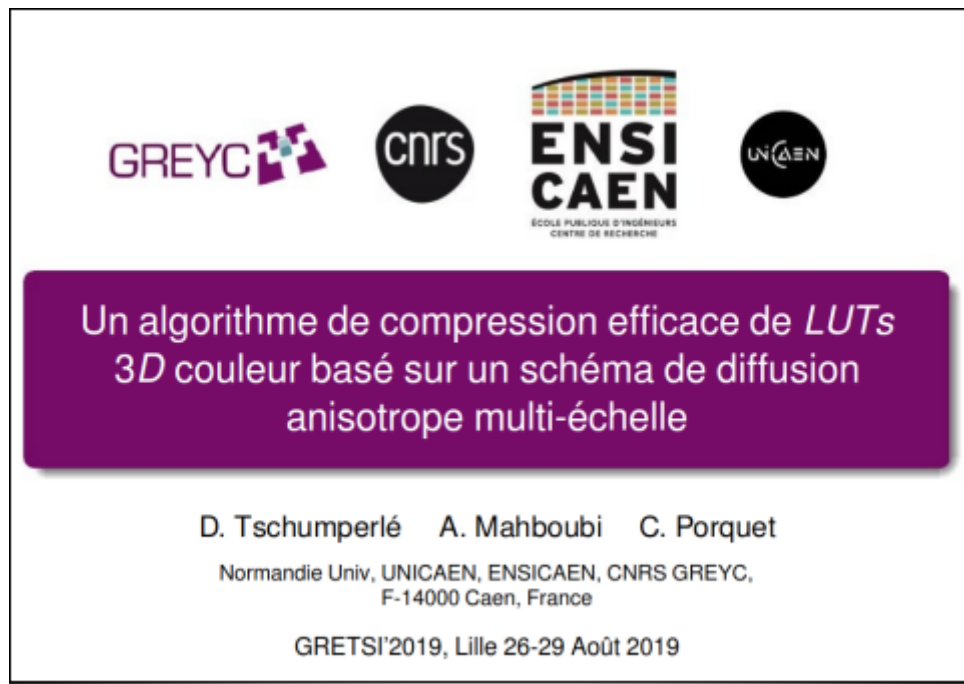


Fig. 4.4 : transparents de présentation des algorithmes de compression/décompression de CLUT (ici, en français).

Notons, pour finir avec ce sujet, que nous avons mis en ligne [une implémentation libre](#) de cet algorithme de décompression de CLUT, en C++ (avec 716 CLUT incluses). Des [discussions ont été également amorcées](#) pour une intégration potentielle dans un module « Color LUT » du logiciel libre [Darktable](#) de développement de photos RAW.

5. Créer des palettes de couleurs par mélange de nuances

Parlons maintenant du filtre récent « **Colors / Colorful Blobs** » qui, ne le cachons pas, s'inspire directement du concept original de [« Playful Palette »](#) imaginé par l'équipe d'*Adobe Research* en 2017, et qui se destine aux artistes illustrateurs (dessinateurs et peintres numériques). L'objectif est de créer des palettes de couleurs, qui ne contiennent que quelques nuances principales (que l'on souhaite utiliser dans une illustration), mais aussi des ensembles de couleurs intermédiaires entre ces nuances, sous la forme de dégradés de couleurs. En piochant uniquement dans les couleurs présentées sur une telle palette, l'artiste préserve plus facilement la cohérence des teintes qui apparaissent dans son œuvre.

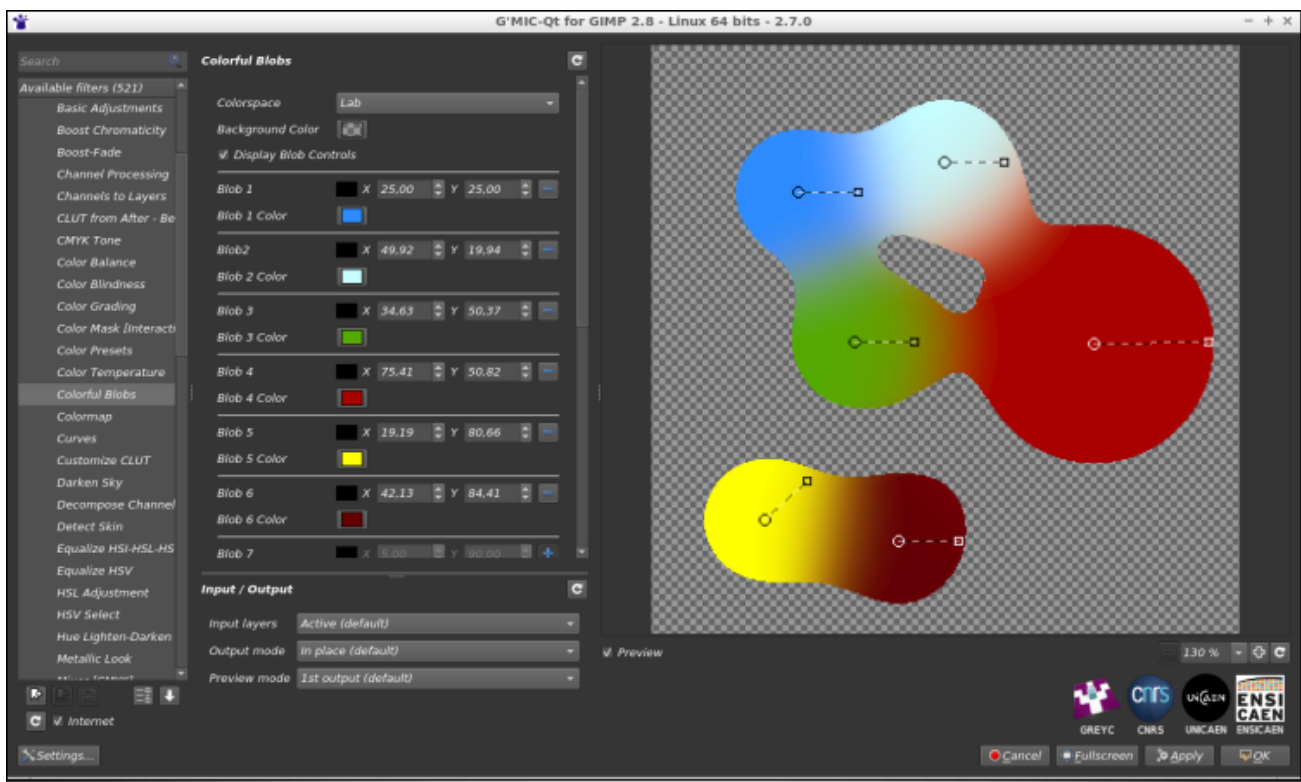


Fig. 5.1 : le filtre « Colors/Colorful Blobs » permet de créer des palettes de couleurs personnalisées, par mélange de nuances.

Comme illustré sur la figure ci-dessus, le filtre propose à l'utilisateur de déplacer des « blobs » colorés qui, en fusionnant les uns avec les autres, créent les dégradés de couleurs désirés. Le résultat du filtre est une image que l'artiste peut utiliser par la suite comme palette, pour piocher les couleurs dont il a besoin.

D'un point de vue purement technique, on utilise une technique de [Metaballs](#)^W (en 2D) pour la modélisation des taches de couleurs. On peut ajouter jusqu'à douze « blobs » distincts et choisir différents espaces de couleur (sRGB, RGB linéaire ou Lab) pour le calcul des dégradés. Le filtre exploite également les fonctionnalités récentes apportées au greffon G'MIC-Qt qui enrichissent l'interactivité utilisateur avec la fenêtre de prévisualisation ([fonctionnalité que nous avons évoquée dans une précédente dépêche](#)), tel que l'on peut le voir avec l'animation ci-dessous, montrant le filtre en cours d'utilisation directement depuis la fenêtre de prévisualisation du greffon (voir aussi cette [vidéo](#), plus longue).

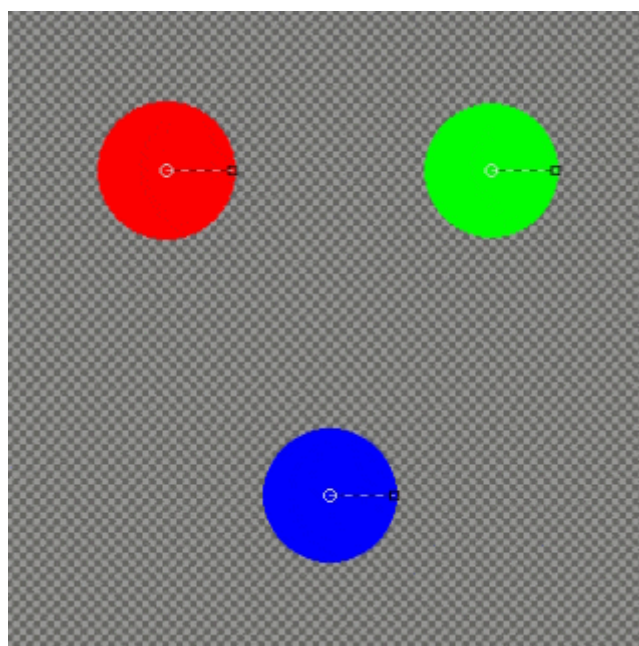


Fig. 5.2 : illustration de l'interaction utilisateur avec le filtre de création de palettes de G'MIC, basée sur la création et le déplacement de « blobs » colorés.

Alors certes, ce filtre ne sera pas utile pour la plupart des utilisateurs de G'MIC. Mais avouez que c'est plutôt *fun*, non ?

6. Il y en a un peu plus, j'vous le mets quand même ?

Décrivons maintenant une sélection de quelques autres filtres et effets apparus durant l'année, peut-être un peu moins originaux que les précédents (pas forcément moins utiles !) mais qui témoignent d'un certain dynamisme du développement de G'MIC.

- Tout d'abord, le filtre « **Rendering / Symmetric 2D Shape** » est d'une aide précieuse quand on souhaite dessiner des figures géométriques possédant des symétries angulaires.

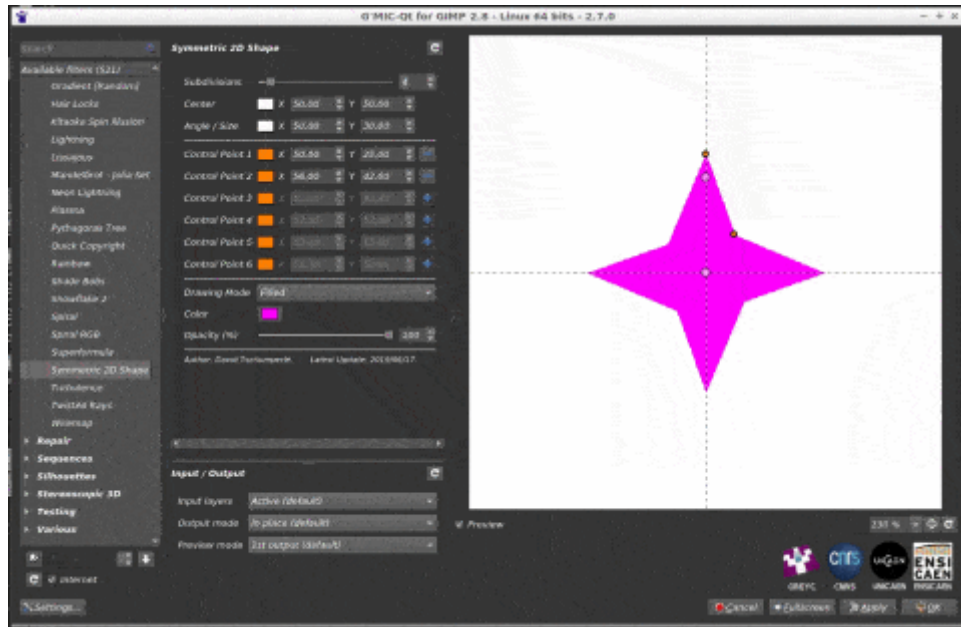


Fig. 6.1 : le filtre « **Rendering / Symmetric 2D Shape** » en action, dans le greffon G'MIC-Qt.

On peut subdiviser le plan jusqu'à trente-deux portions angulaires, chacune de ces portions pouvant accueillir au maximum six point-clés pour définir un profil de forme, ce qui permet de réaliser des rendus de formes potentiellement complexes et variées (tel le super-[shuriken](#)^W ci-dessous ☺).

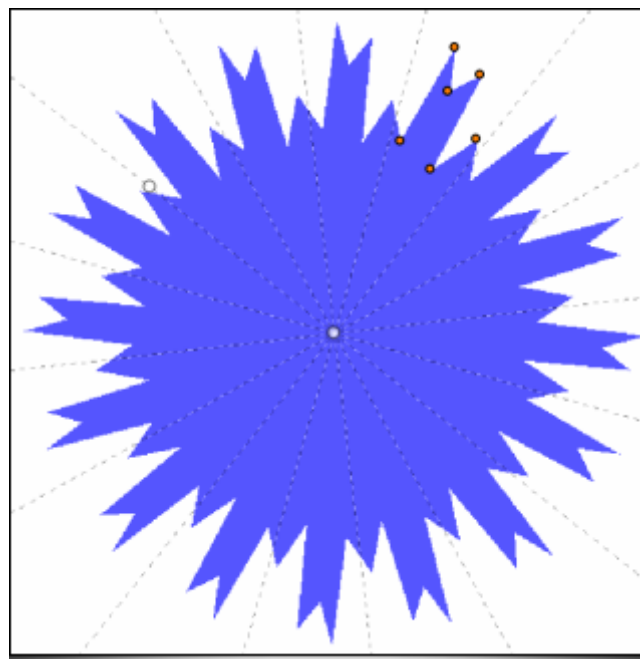
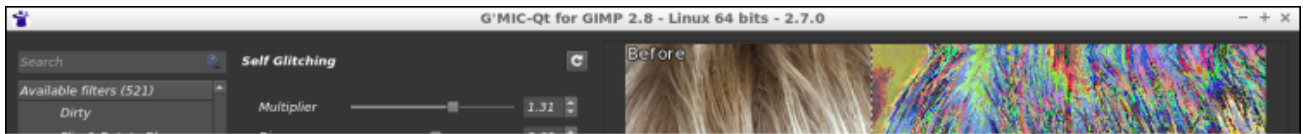


Fig. 6.2 : exemple de figure symétrique complexe réalisable avec le filtre « **Rendering / Symmetric 2D Shape** ».

- Le filtre « **Degradations / Self Glitching** » combine une image avec une version décalée d'elle-même, pour créer un effet de type [Glitch art^w](#). On a le choix entre plusieurs opérations de combinaison bit à bit (*Add, Mul, And, Or, Xor...*) et on peut régler la direction et l'amplitude du décalage, ainsi que divers autres paramètres de contrôle.



*Fig. 6.3 : le filtre « **Degradations / Self Glitching** » aide à ruiner vos photos en toute simplicité !*

Là encore, ce n'est pas un filtre qu'on utilisera forcément tous les jours ! Mais il a le mérite d'exister, et il a d'ailleurs été ajouté en réponse à une demande d'un utilisateur.

- Dans le même style, le filtre « **Degradations / Mess With Bits** » applique certaines opérations sur les valeurs des pixels, vus comme des nombres binaires qu'il ne faut pas hésiter à triturer (décalage et inversion de bits par exemple). Toujours dans l'idée de simuler du *Glitch art*, bien entendu !

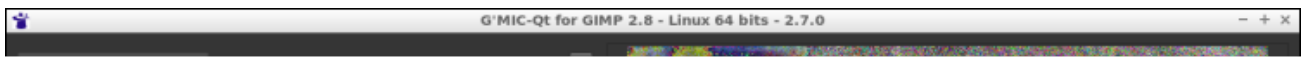


Fig. 6.4 : le filtre « **Degradations / Mess With Bits** », ou comment transformer un adorable bambin en un monstre pustulant...

- Le filtre « **Degradations / Noise [Perlin]** » implémente la génération du [bruit de Perlin^W](#), un modèle de bruit très classique en synthèse d'images, utilisé notamment pour la génération de cartes d'élévations de terrains virtuels. On propose ici une version multi-échelle de l'algorithme original, avec jusqu'à quatre échelles de variations simultanées.



Fig. 6.5 : le filtre « **Degradations / Noise [Perlin]** » propose une implémentation multi-échelle du bruit de Perlin (illustré ici avec deux échelles de variations).

- Le filtre « **Frames / Frame [Mirror]** » est aussi un effet réalisé « sur mesure », pour combler les besoins d'un utilisateur du greffon *G'MIC-Qt*. Ce photographe souhaitait pouvoir redimensionner ses photos pour obtenir un ratio précis *largeur/hauteur*, mais sans avoir à rogner ses images. La solution souhaitée était au contraire de rajouter des informations aux bords de l'image, par symétrisation, pour arriver au ratio désiré. C'est donc ce qu'accomplit ce filtre.



Fig. 6.6 : le filtre « **Frames / Frame [Mirror]** » rajoute des pixels aux bords en symétrisant l'image.

- Notons enfin l'arrivée prochaine d'un nouveau filtre de [débruitage d'images](#)^w avancé, dont le développement est en cours, par [Iain Fergusson](#). Cela fait déjà plusieurs années que Iain contribue à *G'MIC* en expérimentant et proposant des filtres de débruitage originaux, et sa dernière réalisation semble vraiment très aboutie, avec des résultats prometteurs. [Cette vidéo](#) permet de voir ce filtre en action, et d'en apprendre un peu plus sur son fonctionnement.

Maintenant que cet inventaire des nouveaux filtres et effets notables de *G'MIC* se referme, il nous semble important de rappeler que, comme dans beaucoup de projets informatiques, cette partie visible de l'iceberg cache un ensemble de développements plus « bas niveau » (lui, invisible) réalisés au fil des jours afin d'améliorer les possibilités d'interaction du greffon *G'MIC-Qt*, ou encore les performances du langage de script interne ([le langage *G'MIC*](#)), avec lequel tous ces filtres et effets sont implémentés. Ces améliorations ou optimisations incrémentales de la base de code profitent à tous les filtres (même ceux déjà disponibles depuis plusieurs années) et représentent en pratique la majorité du temps de développement. Chers utilisateurs, ne vous étonnez donc pas si aucun nouveau filtre n'apparaît pendant un certain temps. C'est qu'on est sûrement en train d'ajouter des choses indispensables dans le cœur du logiciel !

7. Autres faits notables dans l'évolution du projet

Nous recensons ici d'autres nouvelles importantes qui ont rythmé la vie du projet, depuis août dernier.

7.1. Acceptation des dons

C'est une nouvelle essentielle pour nous : depuis mars 2019, le projet *G'MIC* est habilité à [récolter des dons](#) (par exemple via *Paypal*), pour aider à sa maintenance et son développement !

C'est une bonne chose, car jusqu'à présent, nous n'avions pas de mécanisme simple à proposer, pour qu'un laboratoire de recherche public comme le GREYC puisse accepter des dons, en vue de soutenir le développement

d'un logiciel libre tel que *G'MIC*, pourtant utilisé quotidiennement par plusieurs milliers de personnes à travers le monde (on dénombre entre 400 et 800 téléchargements quotidiens, rien que depuis le site du projet). Et nous n'avons d'ailleurs aujourd'hui aucune autre piste tangible pour financer ce logiciel, à long terme.

Ainsi, nous nous sommes associés à [LILA](#) (*Libre comme l'Art*), association loi 1901 qui accepte les dons, et les reverse au laboratoire sous forme de contrats de collaboration, pour le développement de *G'MIC*.

Fig. 7.1 : Logo de l'association LILA, qui récolte les dons pour le projet G'MIC.

C'est quelque chose qui a été en pratique un peu long à mettre en place, mais maintenant que le système est fonctionnel, on espère bénéficier de ces dons dans le futur pour aider le projet à se développer encore plus vite (l'utilisation possible des fonds est détaillée sur [la page de donations](#), ceci étant bien sûr très dépendant des dons recueillis).

Par souci de transparence, nous [affichons les montants récoltés](#) sur le site Web du projet. C'est nouveau, on ne sait pas vraiment à quoi on peut s'attendre, donc on va voir comment ces dons évoluent. On remercie bien sûr toutes les personnes qui ont déjà participé (ou qui projettent de le faire) à financer notre logiciel libre pour le traitement des images. Notre rêve ultime serait de pouvoir un jour affirmer que l'illustration ci-dessous n'est plus qu'un lointain souvenir !

Fig. 7.2 : la dure réalité du développement du projet G'MIC © ([illustration issue du site CommitStrip](#)).

7.2. Intégration de la « Colorisation intelligente » dans GIMP

Mentionnons également le travail de [Jehan](#), bien connu des lecteurs de LinuxFr.org, de par ses dépêches régulières et ses contributions au développement du logiciel GIMP. Jehan a été recruté par le laboratoire GREYC en septembre 2018 (CDD de 12 mois), pour travailler sur *G'MIC*, grâce à un financement d'ingénieur de développement alloué par l'[institut INS2I du CNRS](#) (qu'au passage, nous remercions chaleureusement).

Une de ses premières missions a été de ré-implémenter l'algorithme de « Colorisation intelligente » de *G'MIC* (que nous avons [déjà présenté précédemment](#)) pour en faire un nouveau mode interactif intégré à l'outil « Remplissage » de GIMP.

Fig. 7.3 : l'algorithme de colorisation intelligente de G'MIC, maintenant disponible dans GIMP, aide les illustrateurs à coloriser leurs dessins plus rapidement.

Jehan a décrit tout son travail dans [un article de blog](#), dont on ne peut que conseiller la lecture. L'idée ici n'est pas de reproduire son texte, mais de faire mention de cette activité, et de la considérer comme une autre contribution originale du projet *G'MIC* aux outils libres pour la création graphique : au laboratoire GREYC, nous sommes réellement heureux et fiers d'avoir imaginé et élaboré un algorithme de colorisation d'images, dont les artistes peuvent se servir via un outil bien intégré dans un logiciel aussi populaire que GIMP.

Cet algorithme de colorisation intelligente a fait l'objet de [publications scientifiques](#), de présentations aux conférences *GRETSI'2017*, *EuroGraphics VMV'2018*, ainsi qu'au [Libre Graphics Meeting 2019](#). Jehan a par ailleurs été interviewé dans [l'émission « Libre à vous »](#), sur Radio Cause commune. Et on a même eu le droit à un petit encart dans un numéro de *Sciences & Vie Junior* !

Fig. 7.4 : extrait du magazine Sciences & Vie Junior où l'algorithme de colorisation intelligente de G'MIC est cité.

Et c'est avec une grande satisfaction que nous constatons que cet algorithme est réellement utilisé, à gauche et à droite, pour diverses réalisations (comme dans [cette chouette vidéo](#) de *GDQuest*, pour la colorisation de *sprites* de jeux vidéos). La recherche scientifique réalisée en laboratoire, qui se démocratise de cette façon, et qui devient accessible au grand public, c'est ça qu'on veut voir !

7.3. Brèves diverses liées au projet

- Récemment, une grosse amélioration des performances de *G'MIC* sous Windows a été accomplie, par le recodage du générateur de nombres aléatoires (maintenant [réentrant^w](#)) et la suppression de [Mutex^w](#) qui créaient un goulot d'étranglement des performances pour tous les filtres nécessitant des séries de nombres aléatoires (et il y en avait beaucoup!). C'était tout bête, mais il fallait y penser, et certains filtres se voient accélérer d'un facteur quatre à cinq sous Windows!
- Depuis décembre 2018, notre greffon *G'MIC-Qt* est utilisable avec le logiciel [Paint.net^w](#), gratuit (non libre) d'édition graphique sous Windows. Ceci, grâce au travail de [Nicholas Hayes](#) qui a écrit le [code glu^w](#) permettant l'interaction entre *G'MIC-Qt* et le logiciel hôte *Paint.net*. Les utilisateurs de ce logiciel peuvent donc maintenant profiter eux aussi des nombreux filtres proposés par *G'MIC*. Cette version du greffon, [disponible ici](#), a déjà été élu « meilleur greffon de l'année 2018 » par les membres du forum de *Paint.net*! ☺ Nicholas surveille les sorties de *G'MIC* de près et a déjà actualisé sa variante du greffon, en version **2.7.0**.
- Depuis octobre 2018, le greffon *G'MIC-Qt* pour GIMP est compilé et proposé pour MacOS, par un nouveau mainteneur, [Andrea Ferrero](#), qui est par ailleurs le développeur principal du logiciel libre [Photoflow](#), un éditeur d'images non destructif ([plus d'informations ici](#)).
 - Depuis la fermeture du réseau social *Google+* nous avons ouvert deux nouveaux comptes, sur [Framasphere](#) et [Reddit](#), pour partager des nouvelles de la vie du projet (mais notre [flux twitter](#) est toujours celui qui reste le plus actif).
- Remercions au passage le père Noël, qui l'année dernière nous a gentiment livré une version matérialisée de notre mascotte « Gmicky ». C'est presque ça! ☺

Fig. 7.5 : la mascotte « Gmicky », apportée par le père Noël, en décembre 2018.

- Le projet *G'MIC* a été présenté lors du [FÊNO](#), le Festival grand public de l'« Excellence Normande », du 12 au 14 avril 2019, au parc des Expositions de Caen. Nous étions hébergés sur le stand du [CNRS Normandie](#), et nous avons réalisé entre autres des démonstrations de [transfert de style \(teaser\)](#) et [d'illumination automatique de dessins au trait \(teaser\)](#).

Fig. 7.6 : nous étions présents au stand du CNRS, pour des démonstrations de G'MIC, à la « Fête de l'Excellence Normande 2019 ».

- Et pour creuser encore plus, quelques autres liens externes que nous avons trouvé intéressants, et qui mentionnent *G'MIC* :
 - une [vidéo de présentation du greffon G'MIC-Qt](#), par *Chris'Tutorial*;
 - la chaîne Youtube [MyGimpTutorialChannel](#) propose des tas de vidéos montrant l'utilisation de *G'MIC-Qt* sous GIMP pour réaliser divers effets (en allemand);
 - [The Clinic](#), journal hebdomadaire chilien a apparemment utilisé *G'MIC* [pour réaliser un effet sur l'une de ses couvertures](#) (via le filtre de lissage « **Artistic / Dream Smoothing** »);
 - un autre [tutoriel vidéo](#), montrant comment utiliser le filtre « **Artistic / Rodilius** » de *G'MIC* pour créer des photos d'animaux stylisés.

8. Le futur, et promis on arrête !

Quand je dis qu'on arrête, c'est bien sûr de la dépêche dont je parle, pas du développement du projet!

Comme nous avons essayé de le montrer, *G'MIC* continue d'être un projet libre actif, et avec ses onze ans d'existence, suffisamment mature pour être utilisé « en production » (qu'elle soit artistique ou scientifique). Et c'est d'ailleurs ce qu'on observe, à notre grande satisfaction, par les nombreux témoignages d'utilisateurs à travers le monde. Nous n'avons jamais suivi de feuille de route précise pour le développement du projet : les fonctionnalités arrivent en fonction des besoins des développeurs et des utilisateurs (et du temps limité que nous pouvons y consacrer!). En ce moment, on s'intéresse beaucoup aux méthodes de traitement d'images basées sur l'utilisation des réseaux de neurones, et des techniques d'[apprentissage profond^w](#). Il est donc possible que certaines

de ces méthodes soient intégrées dans le logiciel, dans un futur plus ou moins proche (nous avons par exemple déjà du code prototypé qui tourne dans *G'MIC* et qui réalise de l'apprentissage de données images avec [des réseaux de neurones convolutionnels](#)^W, mais on en est encore qu'au stade du prototypage...).

Après onze ans de développement (et même vingt ans, si l'on inclut la bibliothèque *Clmg* sur lequel *G'MIC* se repose), on en arrive à un point où le cœur du projet est, techniquement parlant, suffisamment bien conçu et stable, pour ne pas devoir le réécrire complètement dans les années à venir. Par ailleurs, le nombre de fonctionnalités disponibles dans *G'MIC* couvre déjà une grosse partie des besoins classiques en traitement d'images.

L'évolution de ce projet peut donc emprunter plusieurs chemins, en fonction des ressources humaines et matérielles que nous pourrions y consacrer dans le futur (en développement, mais aussi en gestion de projet, en communication, etc.). Arriver à accroître ces ressources sera sans aucun doute un des enjeux importants de ces prochaines années, si l'on veut que *G'MIC* continue son avancée (et ce ne sont pas les idées qui nous manquent!). Dans le cas contraire, il pourrait arriver que le logiciel soit seulement maintenu à son état actuel (et fonctionnel).

C'est, bien sûr, dans une optique de progression, que nous avons récemment mis en place [la page de dons](#). Nous espérons aussi que d'autres opportunités se présenteront prochainement pour nous permettre de valoriser ce logiciel.

Nous en profitons, pour relancer un appel à candidature pour [un poste d'ingénieur d'aide au développement de G'MIC](#), dont nous avons obtenu le financement pour l'année universitaire 2019-2020. Si vous souhaitez plonger avec nous dans cette belle aventure du traitement d'images libre, tout en profitant du climat tropical de notre rayonnante ville de Caen, cette offre est faite pour vous! (et il faudrait se dépêcher de répondre, car l'annonce expire le 28 août ☺).

Voilà, cette dépêche est maintenant terminée, merci d'avoir tenu jusqu'au bout, vous pouvez reprendre une activité normale! J'essayerai autant que possible de répondre aux questions éventuelles dans les commentaires.

Post-scriptum — Notez que l'animation 3D affichée comme *teaser* en tête de dépêche a été effectivement générée par *G'MIC*, via la commande `gmic x_starfield3d`. L'occasion de rappeler que *G'MIC* possède aussi son propre moteur de rendu capable d'afficher des objets 3D simples, ce qui est bien pratique pour la visualisation scientifique! Nous en reparlerons peut-être dans une prochaine dépêche...

Aller plus loin

 [Le projet G'MIC](#) (188 clics)

 [Fil Twitter des News du projet](#) (30 clics)

 [Série d'articles G'MIC sur LinuxFr.org](#) (68 clics)

Traduction

Posté par [Lolo Spit](#) ([site web personnel](#), [adresse XMPP](#)) le 24/08/19 à 09:24. Évalué à 3.

Quid de la traduction ?

Si G'MIC veut se faire une place en entreprise ou chez Mme Michu, il va falloir s'y mettre...

[+] Re: Traduction

Posté par [Ysabeau](#) ([site web personnel](#), [adresse XMPP](#), [Mastodon](#)) le 24/08/19 à 10:12. Évalué à -6.

Re: Traduction

Posté par [Lolo Spit](#) ([site web personnel](#), [adresse XMPP](#)) le 24/08/19 à 11:02. Évalué à 2.

Alors elle peut être pas, mais j'en connais d'autres, avec certitude, oui !
Quitte à payer plutôt que donner.