03/04/17

Traitement du signal et des images

Minitoshop :

Une implémentation partiel d'une application de retouche d'image

Gaëtan Gousseaud
Julie Rymer

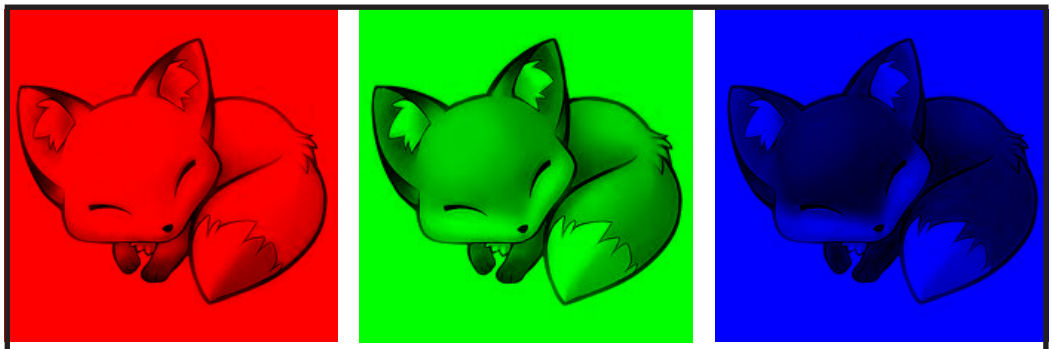
Présentation

Étant de fervants utilisateurs de logiciels de retouche d'images depuis plusieurs années, et comme nous avions chacun envie d'implémenter divers algorithmes de traitement pour comprendre leur fonctionnement, nous avons choisi de faire un logiciel simpliste de retouche d'image. Nous avons essayé d'implémenter le plus de fonctionnalités qu'on retrouve usuellement (agrandissement, rotation, luminosité, *etc*), puis des fonctionnalités d'amélioration d'image (atténuation de bruit, réduction du flou, *etc*) pour finir par un ensemble de traitements plus originaux, avec des modifications d'image au niveau de ses couleurs ou de sa forme, qui peuvent servir dans un but artistique.

Traitement de couleurs basique :

Noir et blanc

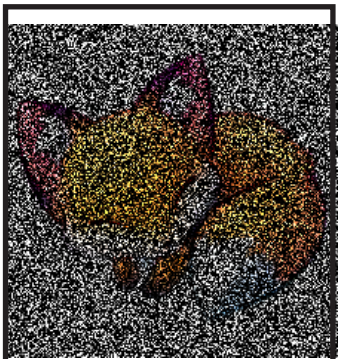
Inverse



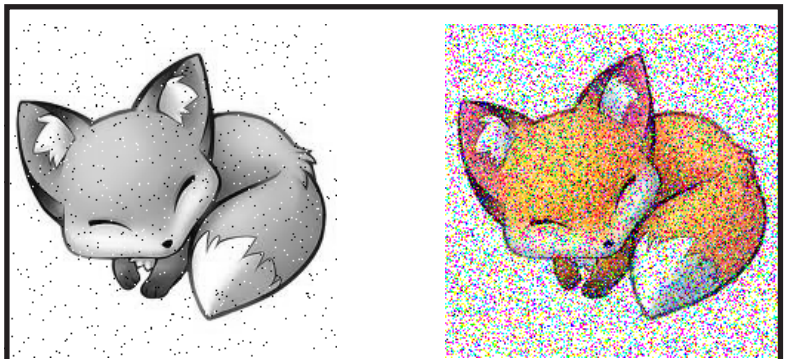
Trois canaux de couleur R V B

Ajout d'un masque de couleur

Bruits :

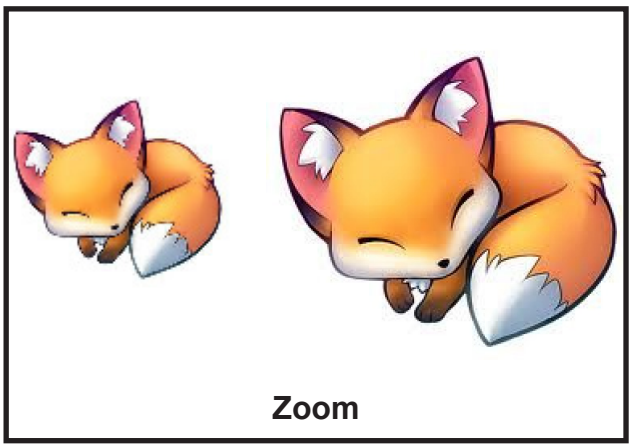


Bruit de Perlin

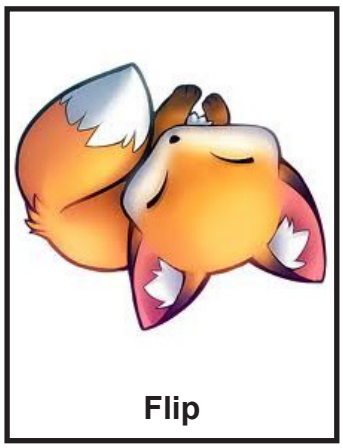


Bruit poivre&sel 0.02 / 0.3

Reformatage :



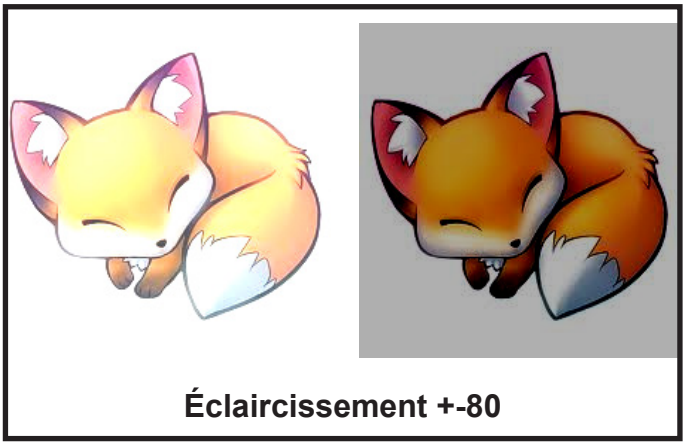
Zoom



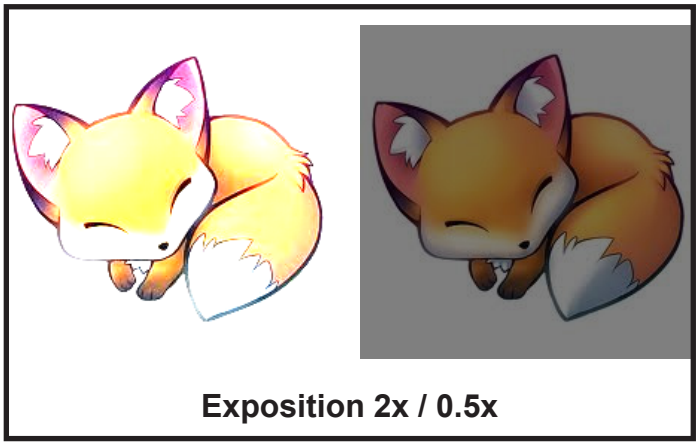
Flip

Rotation

Ajustement de la luminosité :



Éclaircissement +-80

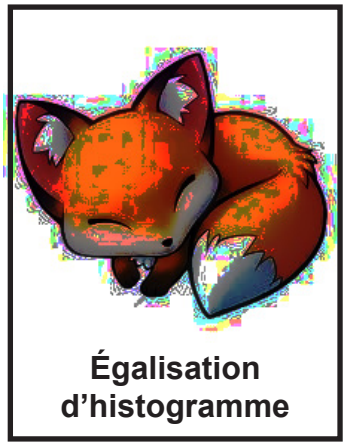


Exposition 2x / 0.5x

Pour ajuster la luminosité il existe plusieurs méthodes. Nous avons choisi celles qui consistent à additionner ou multiplier tous les pixels par une valeur donnée.

Amélioration du constraste :

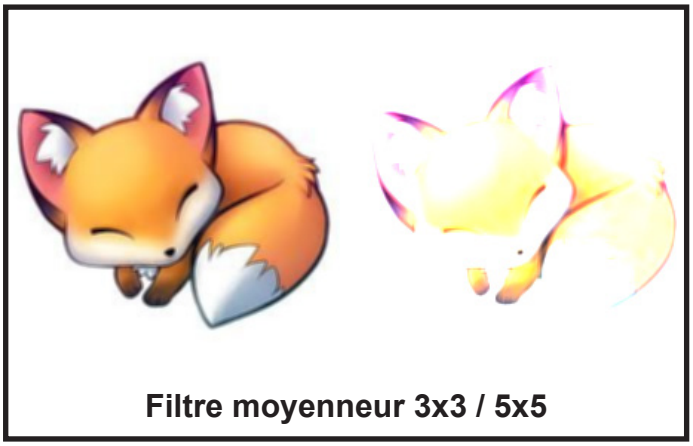
Extension linéaire



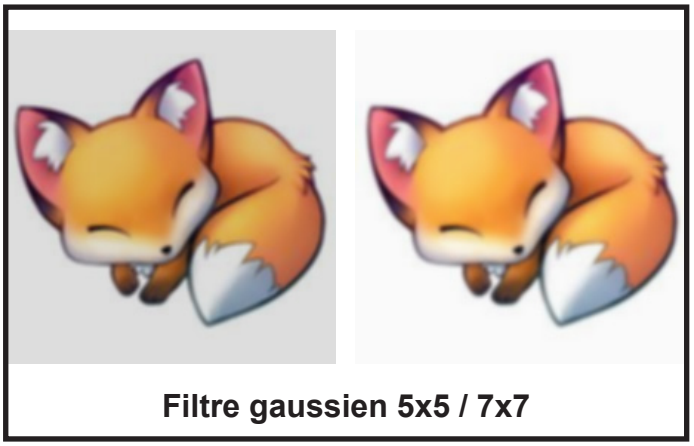
Égalisation d'histogramme

On a ici appliqué les fonctions de traitement de contraste sur l'image après lui avoir changé son exposition à 0.5 (image du dessus). On remarque ici que l'extension linéaire marche très bien, l'image retrouvée est identique à l'original. En revanche, égaliser l'histogramme ne donne vraiment pas un bon résultat sur cette image.

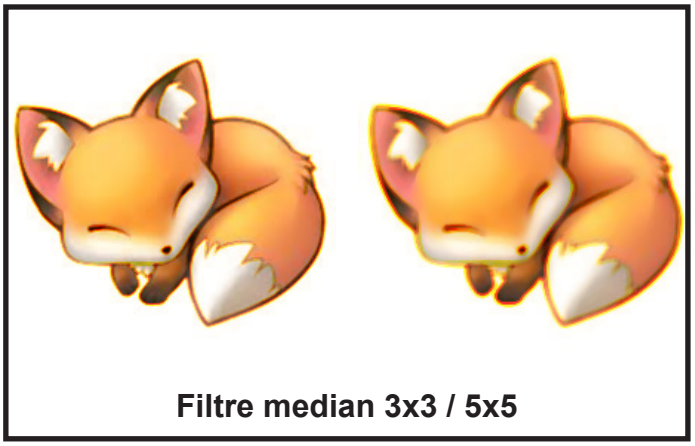
Flous :



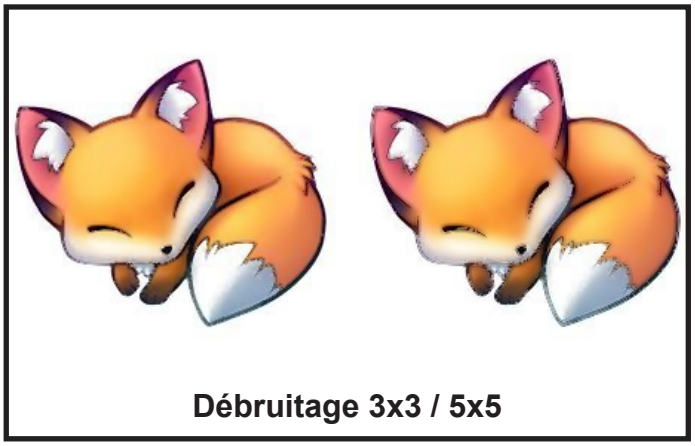
Filtre moyenneur 3x3 / 5x5



Filtre gaussien 5x5 / 7x7



Filtre median 3x3 / 5x5

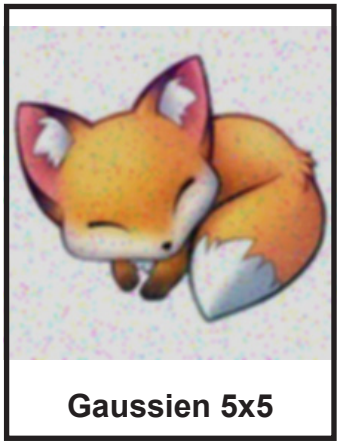


Débruitage 3x3 / 5x5

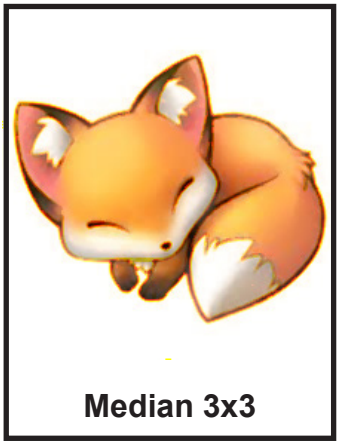
Comme on pouvait s'y attendre, c'est le filtre de débruitage qui abîme le moins l'image à plus fort effet.

Flous appliqués à une image bruitée :

Moyenneur 3x3



Gaussien 5x5



Median 3x3

Débruitage 5x5

Ici c'est plutôt le filtre médian qui s'en sort le mieux, suivi du filtre de débruitage qui a le meilleur résultat de tous si l'on ne tient pas compte du fond. Ce résult at est rassurant étant donné que c'est le rôle même du filtre de débruitage. Au lieu d'appliquer le même traitement à tous les pixels d'une image, Il va uniquement avoir un effet sur les pixels qu'il aura identifiés comme «aberrants» par rapport à la moyenne et l'écart-type de ses voisins.

Contours :

Filtre de netteté 3x3

Filtre de netteté 3x3 après un moyenneur 3x3

Le filtre de netteté est un filtre auquel on a soustrait le filtre moyenneur. De ce fait il accentue tous ce que le filtre moyenneur aurait enlevé, les contours, ce qui cause ce défloutage. En le testant après avoir appliqué un filtre moyenneur de même taille, on voit qu'il nous donne une image proche de l'image original.

Filtre de Sobel vertical droite

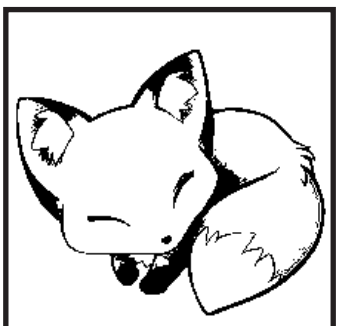
Addition des quatre filtres de Sobel 3x3

Filtre de Laplacien

Réduction des bits d'encodage :

Noir&Blanc encodé sur 3bits

Couleur encodé sur 2bits

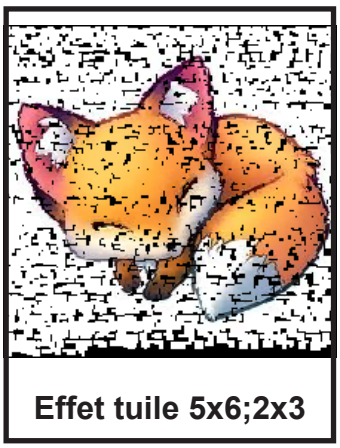


Seuillage à 128 ou encodage sur 1 bit du Noir&Blanc

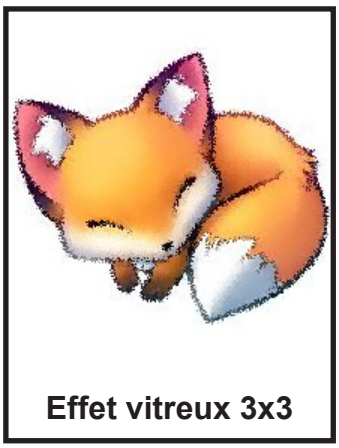
Seuillage à 128 ou encodage sur 1 bit de la couleur

On peut réduire le nombre de bits sur lequel est codé un pixel, pour cela il faut récupérer le nombre de niveaux possibles sur ce nouvel encodage ($2^{nombre\_de\_bits}$), donner une valeur à ces niveaux possibles, puis calculer le niveau le plus proche de la nouvelle valeur du pixel par division. On remarque que lorsque l'on change l'encodage sur 1 bit, les valeurs ne vaudront plus que 0 ou 255, ce qui donne le même résultat que si l'on avait fait un seuillage à 128.

Déformations de couleurs :

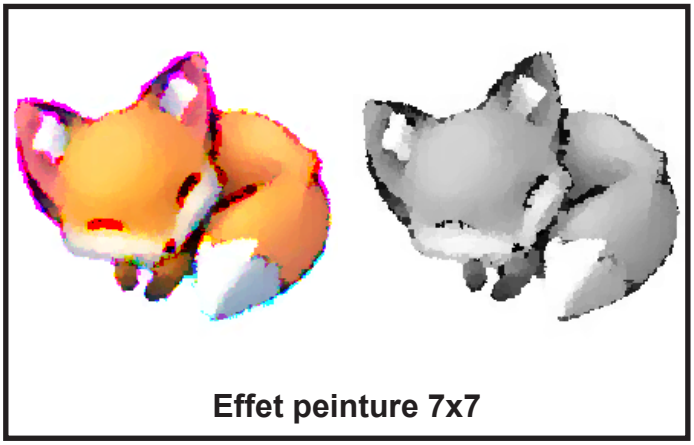


Effet tuile 5x6;2x3

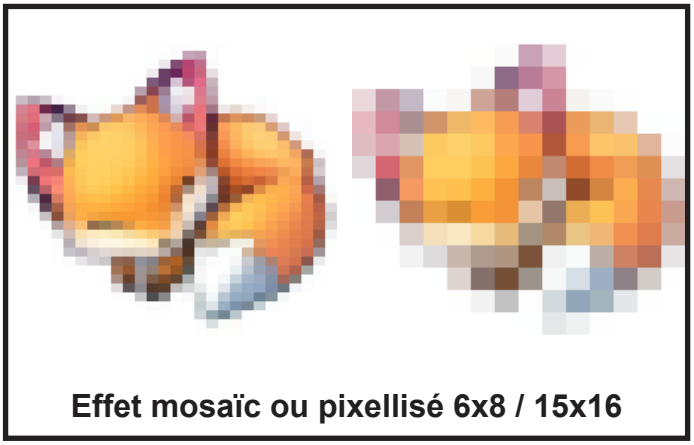


Effet vitreux 3x3

L'effet tuile va afficher l'image originale par blocs, dont la taille et l'espacement seront influencés par le hasard. L'effet vitreux va, lui, pour chaque pixel, prendre au hasard sa valeur parmis la sienne ou celle de ses voisins.

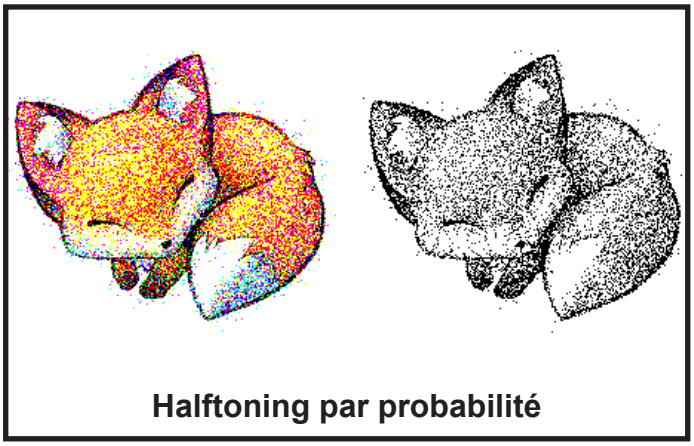


Effet peinture 7x7

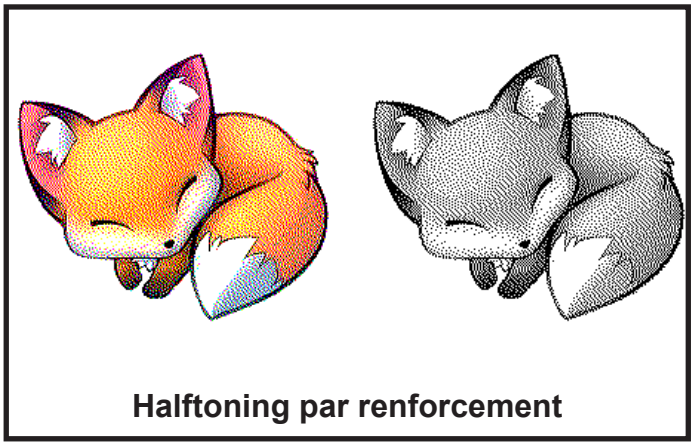


Effet mosaïc ou pixellisé 6x8 / 15x16

L'effet de peinture va attribuer au pixel de la nouvelle image, la valeur la plus présente chez ses voisins dans l'image originale. L'effet pixellisé est obtenu en parcourant l'image bloc par bloc, et en donnant à tout le bloc la valeur moyenne des pixels qu'il contient.



Halftoning par probabilité

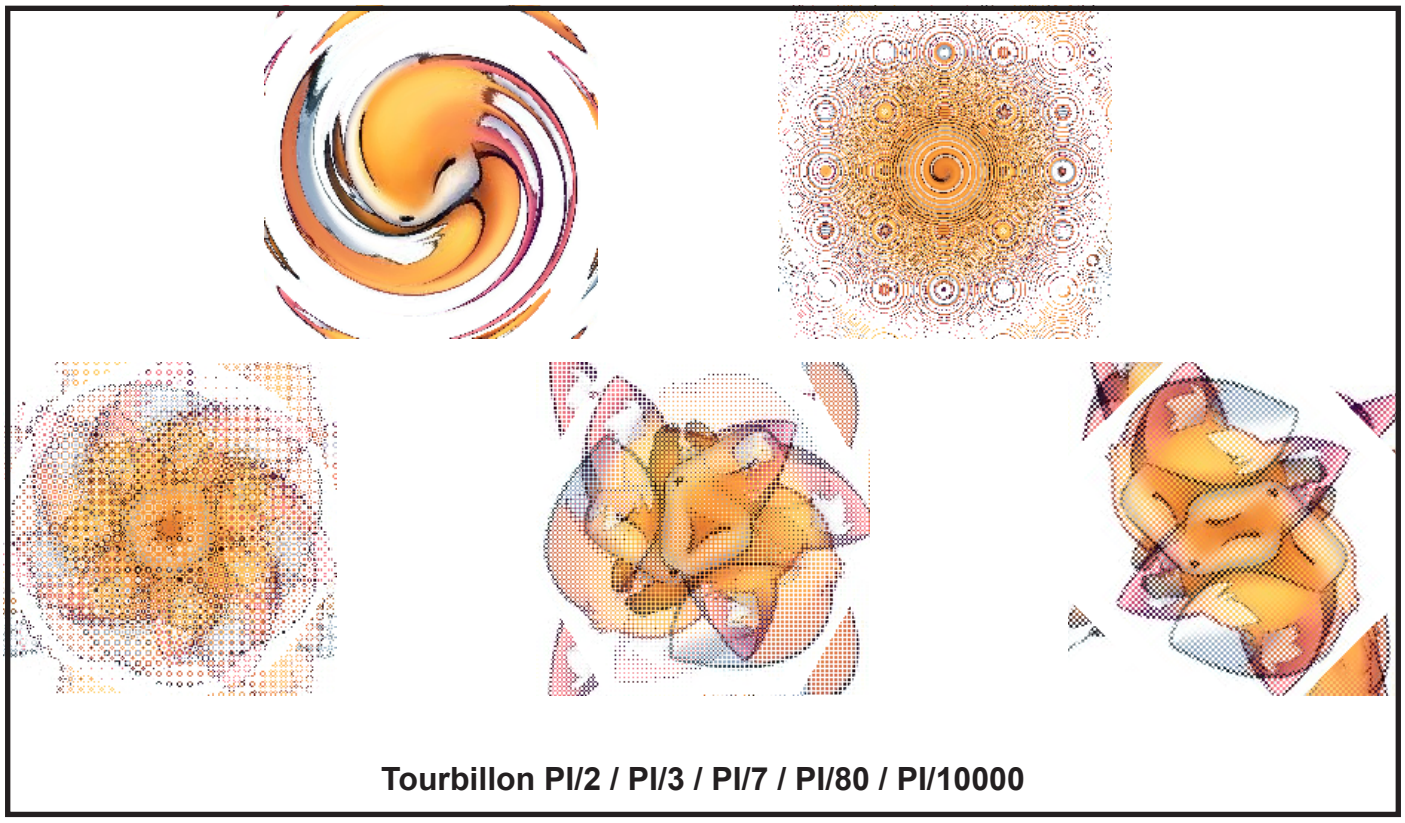


Halftoning par renforcement

Le halftoning consiste à n'afficher une image qu'avec des pixels qui ne contiennent plus que les valeurs 0 ou 255 (monochrome), sans changer les couleurs perçue visuellement. La première méthode consiste à créer un tableau de double random, et, en les comparant avec les valeurs de l'image, plus le pixel aura une grande intensité lumineuse, plus il aura de chance de devenir blanc. La deuxième méthode parcours l'image et mets à 255 le pixel s'il est plus grand que 128, puis mets à jour ses voisins en fonction de la différence entre l'ancienne valeur du pixel, et celle qu'il a reçu (plus un pixel est mis à blanc alors qu'il était foncé, moins ça arrivera à ses voisins).

Déformations géométriques :

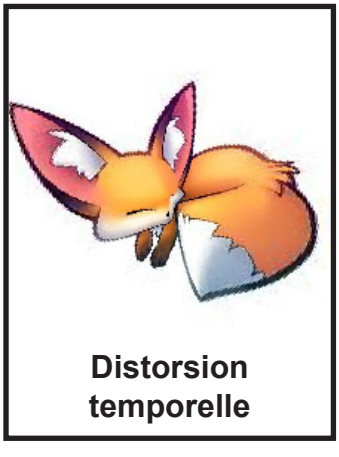
À l'aide des coordonnées polaires, il est possible de faire plusieurs transformations intéressantes.



Tourbillon PI/2 / PI/3 / PI/7 / PI/80 / PI/10000

Le tourbillon récupère le pixel dans l'image original en effectuant une rotation depuis le centre dont l'angle est de plus en plus grand en fonction de la distance.Nous avons beaucoup apprécié cet effet car en jouant sur le coefficient de déformation et en amplifiant l'effet de la distance, nous avons pu obtenir des fractales ainsi que l'illustration de l'illusion d'immobilité qu'on a en regardant un objet qui tourne trop vite (comme les jantes des roues d'une voiture en mouvement).

Fisheye



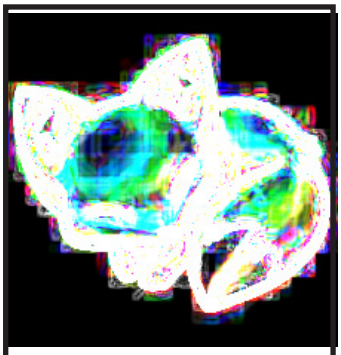
Distorsion temporelle

Le Fisheye et la distorsion temporelle ne change pas l'angle du pixel récupéré mais son radius en fonction de sa distance au centre. Il est réduit pour faire le fisheye, et aggrandit pour la distorsion temporelle.

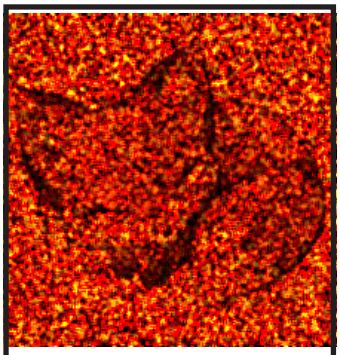
vague x20 verticale / horizontale / croisée

En appliquant une sinusoïde lorsque l'on récupère la couleur du pixel dans l'image originale, on obtient la représentation visuelle d'une fonction sinusoïdale par des vagues. Les vagues sont dirigées à la verticale, à l'horizontale ou dans les deux directions en fonction de si l'on applique la fonction à l'abscisse, l'ordonnée ou les deux.

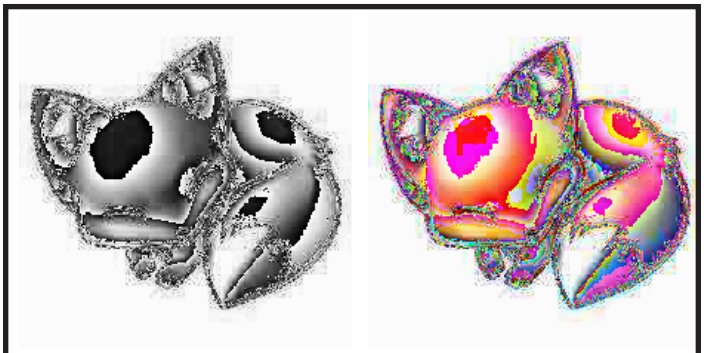
Bugs visuels amusants :



Addition des quatre filtres de Sobel 5x5



Bruit de Perlin puis filtre médian 3x3



Changement d'exposition 5x avec les valeurs des pixels qui bouclent entre 255 et 0 au lieu de saturer

Au cours de nos manipulation, nous sommes aussi parfois parvenu, par des ratés, à des résultats aux effets visuels intéressants.

Conclusion

Avec ce projet, nous avons pu voir à quoi ressemblait un logiciel de programmation à vocation mathématique ainsi qu'avoir un aperçu du fonctionnement des logiciels de traitement d'images usuels (photoshop avec ses nombreux filtres déformants par exemple). Cela nous a aussi donné une occasion d'avoir une vraie démonstration visuelle de principes mathématiques plutôt que de seulement les appliquer sur des choses abstraites. Il a aussi été intéressant de constater les différences de performance d'un langage à un autre en fonction des tâches demandées. Si Scilab est assez lent pour parcourir des boucles, il est en revanche très rapide pour tout ce qui est manipulation de matrice (extraction de sous-matrice, création d'une grande matrice de nombres aléatoires, *etc*), il nous a donc fallu nous adapter au langage et en utiliser les outils le plus possible pour avoir de meilleurs temps de traitement (comme insérer directement une sous-matrice de couleurs plutôt que de faire une boucle supplémentaire qui traite chaque couleur séparément).