ENSiiE FIP-FIPA1A 2018
Algorithmique et Programmation

Algorithmique et Programmation

TP Tableaux 2D : manipulation d'images

1) Mise en place

- Assurez vous que vous disposez du support de cours sur les tableaux 2D et les manipulations élémentaires d'image : IPI-CM-Image.pdf
- Récupérez les fichiers source à compléter.
- Pour vous assurer que tout est en ordre (la première image est l'image à traiter, la deuxième est créée par le programme, c'est le résultat du traitement) , dans le terminal exécutez la commande :

```
$ python tp_image.py test.pgm res.pgm
```

2) Travail à réaliser

Vous devrez rajouter des fonctionnalités à l'application dont le squelette vous a été fourni, et qui ne comporte initialement que 2 fonctions de transformation : inversion vidéo (négatif), et ajout de bruit (dans le but de tester 2 approches de correction de défauts d'images). Vous devrez aussi modifier le programme principal afin qu'il fasse appel aux nouvelles fonctions que vous avez mises en oeuvre.

Fonctions à intégrer dans le module effets_photom.py

- Fonction d'éclaircissement d'image : `plusClairNB`. Cette fonction doit rendre votre image plus claire en augmentant chaque niveau de gris de 25%.
- Fonction de création d'un effet de relief : `deriv1xNB`, chaque pixel est obtenu en effectuant la différence entre son niveau de gris et celui du pixel qui se situe immédiatement à sa gauche. La différence obtenue peut prendre des valeurs entre -255 et 255. On tronquera les valeurs en dehors de l'intervalle [-128,127], qu'on ramènera entre 0 et 255 pour être affichées.

Fonctions à intégrer dans le module effets_geom.py

- Fonction de création d'un effet de « fonte » : `effetFonteNB`. Cette fonction traite N pixels choisis aléatoirement. En général on prend N égal au nombre de pixels. Chaque pixel sélectionné est comparé à celui situé immédiatement sur la ligne inférieure. Si le pixel du bas est plus clair, alors il prend la valeur du pixel au dessus.
- Fonction de réduction de la taille d'une image : `quartImageNB`. Cette fonction calcule une nouvelle image dont les dimensions sont les dimensions initiales divisées par 2, et les pixels sont obtenus en ne retenant qu'une ligne sur 2, et sur chaque ligne, une colonne sur 2.

Fonctions à intégrer dans le module filtrage.py

- Fonction de correction du bruit présent dans une image : `filtrerMedianImageNB`. Cette fonction permet de remplacer la valeur d'un pixel par la valeur médiane des pixels de son voisinage. Elle est calculée en mettant dans un tableau toutes les valeurs au voisinage d'un pixel (fenêtre carrée de taille impaire centrée sur le pixel). Le tableau est ensuite trié, la valeur médiane est celle au milieu du tableau. Pour éviter de faire des tests sur chaque pixel, on ne pratiquera pas le filtrage des bords de l'image.
- Fonction de lissage d'une image par calcul de la moyenne des pixels sur un voisinage :

`filtrerImageNB`. Cette fonction a pour but de remplacer la valeur d'un pixel par la moyenne des valeurs de ses voisins pris dans une fenêtre carrée de taille impaire centrée sur le pixel à filtrer. Pour rendre votre fonction évolutive, vous passerez le filtre et sa taille en paramètre. Dans le cas d'une moyenne, tous les coefficients du filtre possèdent la même valeur, l'inverse du nombre de coefficients. Faites attention à ce que le filtrage soit applicable, ce n'est pas le cas des pixels proches du bord de l'image. Ajoutez les commentaires au format Doxygen.

Tests de vos fonctions dans le module tpimage.py

- Vous modifierez le module `tpimage.py` pour qu'il propose un menu avec les différentes fonctions disponibles.

Extension : traitement des images couleurs (ppm), à intégrer dans le module effets_photom.py

- Fonction de transformation d'une image couleur en image de gris : `enNiveauxDeGris`. Cette fonction ne s'applique qu'aux images RVB, et transforme chaque pixel en luminance par la formule :

$$L = 0.707 * R + 0.202 * V + 0.071 * B$$

Autres extensions : traitement des images couleurs

- Ecrivez les fonctions `plusClairRVB` et `effetFonteRVB` qui réalisent la même chose que `plusClairNB` et `effetFonteNB` mais sur des images couleur : un triplet (R,V,B) par pixel.