Rapport du TP n°5

Traitement d'images couleur
Changement d'espaces couleur

Données multimédia (HAI605I)
Partie Image

L'objectif de ce TP est de traiter des images couleur au format ppm et de changer d'espace afin d'effectuer des traitements spécifiques.

GARCIA-PENA Loris
L3 Groupe B
github : https://github.com/GarciaPena-Loris/Donnees-Multimedia

1) Obtention d'une image en niveaux de gris à partir d'une image couleur

Le but de cette première partie est dans un premier temps de téléchargé une image ppm et pgm identique. Dans un second temps, d'écrire un programme qui transforme une image ppm en pgm. Pour finalement comparer les deux images pgm à l'aide d'une nouvelle fonction qui calcule d'erreur quadratique moyenne (EQM) :

Premièrement, l'image que nous avons choisi au format ppm et pgm est la suivante :

Monarch.ppm

Monarch.pgm

Dans un second temps, nous allons donc écrire un programme qui transforme n'importe quelle image ppm en pgm. Ce programme repose simplement sur ce principe :
Y = *0.299 R + 0.587 G + 0.114 B*

Ainsi, pour passer d'une image ppm à pgm, il suffit de faire la moyenne des 3 octets d'un pixel et d'enregistrer cela dans notre nouvelle image. Avec notre image, on obtient le résultat ci-conte.

Cependant, il est assez difficile de savoir s'il y a vraiment une différence avec notre image pgm d'origine.

Pour faire cette comparaison, nous allons ainsi créer une fonction qui calcul d'**EQM** de nos images.

L’erreur quadratique moyenne est une mesure de la précision d’un estimateur statistique. Elle calcule la *moyenne* des *carrés* des *écarts* entre les valeurs observées et les valeurs prédites par le modèle. Plus l’erreur quadratique moyenne est faible, plus le modèle est fiable.

Avec nos deux images, on obtient un **EQM** de ‘17.639980’.

2) Transformation de l'espace RGB vers l'espace YCbCr

Cette partie consiste à séparer une image RGB en 3 composantes Y, Cb et Cr

Il faut rappeler que dans une image YCbCr, la composante Y représente la luminance (la luminosité) de l'image, tandis que les composantes Cr et Cb représentent la chrominance (la couleur) de l'image.

Pour ce faire, nous utilisons, de manière analogue à la première partie, une formule mathématique qui permet d'extraire ces composantes à partir d'une image RGB :

$$Y = 0.299\ R + 0.587\ G + 0{,}114\ B$$
$$Cb = -\ 0.1687\ R - 0{,}3313\ G + 0.5\ B + 128$$
$$Cr = 0.5\ R - 0{,}4187\ G - 0{,}0813\ B + 128$$

A partir de note image ppm, on a obtenu les images pgm Y, Cr et Cb suivante :

Monarch_Y.pgm

Monarch_Cb.pgm

Monarch_Cr.pgm

On constate que la composante Y ressemble à note image pgm de base, ce qui est normal étant donné que cette composante représente la luminosité de notre image. Les composantes Cr et Cb sont plus floues et bruitées que la composante Y. On peut également remarqué que sur la composante Cb la couleur (jaune) des ailes du papillons ressorte fortement.

3) Transformation de l'espace YCbCr vers l'espace RGB

Dans cette partie, nous allons faire le travail inverse de l'étape précédente. C'est-à-dire que nous allons recréer une image RGB à partir d'une image YCbCr.

Toujours sur le même principe, nous appliquons les formules suivantes :

$$R = Y + 1.402\ (Cr - 128)$$
$$G = Y - 0.34414\ (Cb - 128) - 0.714414\ (Cr - 128)$$
$$B = Y + 1.772\ (Cb - 128)$$

Ce programme nous donne donc en sortie une image ppm, mais également une image pgm de chaque composante :

Monarch_R.pgm

Monarch_G.pgm

Monarch_B.pgm

Monarch_A.pgm

On remarque que cela correspond bien à note image ppm de départ. On a un EQM de 2.66.

Finalement, on remarque qu'il est assez facile de passer d'images RGB à YCbCr et inversement à l'aide des formules mathématiques correspondantes.

3) Inversion de composantes à la reconstruction

Cette partie nous permet de générer des images avec de nouvelles couleurs originales en changeant l'ordre des R, G et B.

À partir de notre image de base, on peut alors construire 6 nouvelles images :

Monarch_RGB

Monarch_RBG

Monarch_GRB

Monarch_GBR

Monarch_BRG

Monarch_BGR

4) Modification de la luminance d'une image couleur

Cette dernière partie a pour objectif d'augmenter la luminosité d'une image ppm.

Pour ce faire, nous pouvons utiliser nos composantes Y, Cb, Cr. En effet, si on augmente la luminosité de notre composante **Y** et que l'on réassemble notre image RGB, on va obtenir une image plus claire et inversement.

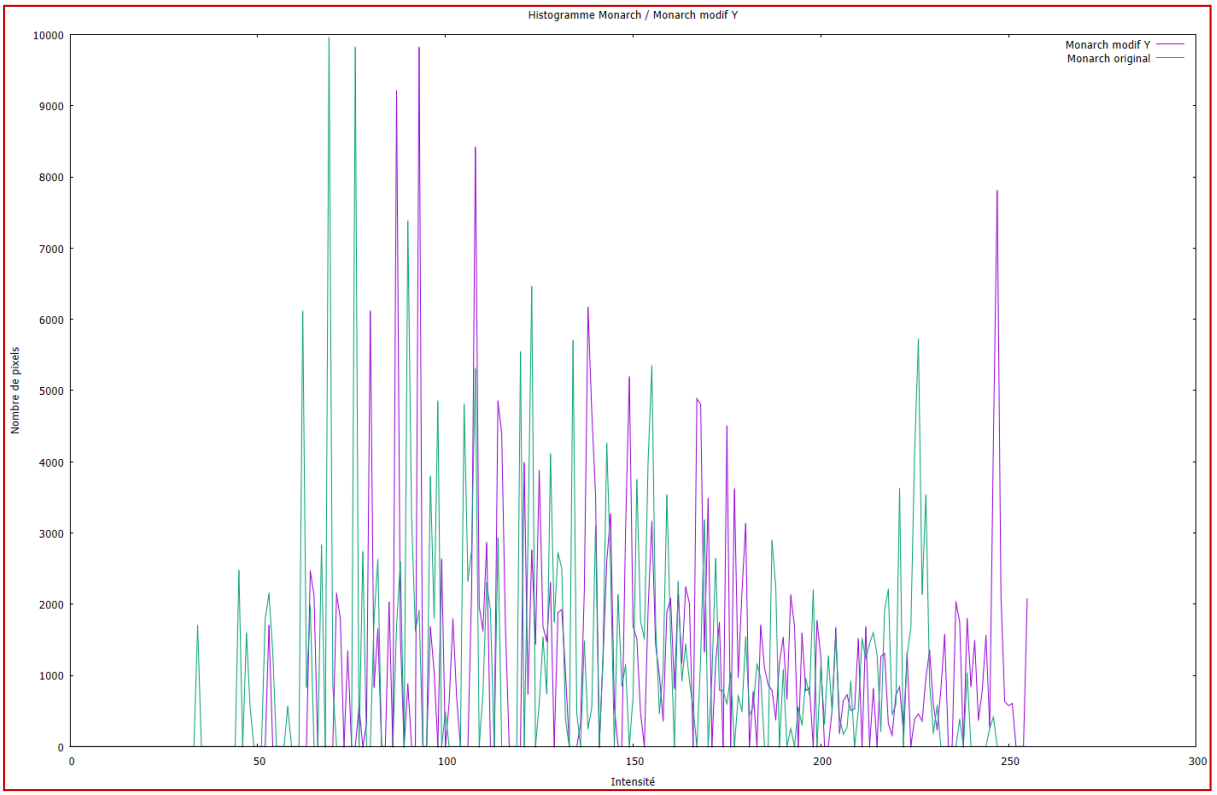
Il faut donc dans un premier temps augmenter la luminosité. Pour faire cela, on reprend le principe des seuils d'une image, mais avec un s (k) compris entre -128 et +128.

Monarch_modifyY.pgm

Une fois que l'on a notre nouvel Y, il suffit de rassembler l'image avec notre fonction **YCbCr**, on obtient alors une nouvelle plus éclairée (dans notre cas, car k > 0) :

Monarch_A_ModifyY.ppm

On peut montrer cette différence de luminosité à l'aide d'un histogramme :



histo-coul-monarch.pdf

On remarque bien que la courbe du nouveau papillon est décalée vers la droite (donc image plus claire) que notre image d'origine.

Finalement, on remarque que notre image est ainsi plus claire qu'auparavant sans pour autant avoir pendu en netteté ou quoique ce soit.

5) Conclusion

Pour conclure, cette partie nous permet de mieux comprendre ce qu'est une image YCbCr et la comparer avec les images RGB que nous connaissions. De plus, nous avons appris une méthode pour modifier à luminosité d'une image sans perdre en qualité.

Pour ma part, j'ai trouvé ce TP très amusant et assez simple. C'est drôle de pouvoir modifier la couleur d'une image aussi simplement. De plus, je souhaite faire la partie BONUS, mais dans un second temps, car là ce sont les vacances 🙃.