Rapport IMG TP 2

Définition et résolution

Definition: 500x580 => 290000px

x et y correspondent aux coordonnées du premier point du rectangle avec w sa largeur et h sa hauteur

Resolution horizontale/verticale: 4px/mm

Taille d'un pixel: 0.25x0.25 mm; Taille de l'image: 125x145mm

C'est devenu illisible, certains détails ont été perdus, l'hypothese serait que 3 pixels sur quatre aient été perdus

Nouvelle resolution: 2px/mm, Taille d'un pixel: 0.5x0.5 mm ,Taille de l'image: 125x145mm

On ne retrouve pas l'image de départ, les pixels sont dupliqués pour créer les pixels manquants, créant des groupes de 4 pixels identiques

Avec les mêmes manipulations, on perds encore plus de détails, et en revenant à une definition « normale » on a toujours la même image peu détaillée, on a maintenant des groupes de 16 pixels identiques

Si on active l'option « Bicubic », on remarque qu'en remettant l'image à une definition initiale, la qualité des détails a legerement diminué mais cela reste plus lisible que sans cette option, notre hypothèse serait que chaque pixel est créé à partir de la moyenne des valeurs des pixels autour, le processus serait similaire lorsqu'on agrandit l'image

Extraction des composantes

Chaque image en niveaux de gris correspond à la valeur RGB de chaque pixel affichée en niveaux de gris.

Combinaison des composantes

On a obtenu en premier l'image c en faisant des substitutions avec les 3 composantes, donc la composante bleue va dans le « channel » rouge, la rouge dans le channel vert et la verte dans le channel bleu. On a remarqué que depuis l'image de base, le bleu devait devenir rouge, le rouge devenir vert et le vert devenir bleu

Pour l'image a, on a simplement omis de remettre la composante bleue dans son channel.On a remarqué que les composantes bleues du cyan et du magenta devaient être retirées.

Pour l'image b, il a suffit de retirer la composante verte de son channel, puis d'intervertir les composantes rouges et bleues. On a remarqué que les composantes vertes devaient être retirées mais que de plus le bleu devait devenir rouge et inversement.

Variation de quantification

Cette opération correspond au fait de garder les 4 bits de poids fort pour chaque pixel, la composante est donc maintenant codée sur 4 bits et peut donc prendre 16 valeurs

Les images semblent à première vue similaires, on remarque cependant quelques différences, notamment un léger jaunissement par endroits, et d'autres endroits sont legerement plus foncés, cela pourrait s'expliquer par le fait que la composante bleue soit plus atténuée que les autres dans son codage.

On a un gain théorique de 8bits par pixel sans grandement modifier l'aspect de l'image originale.

Sous-echantillonage

Cette opération nous fait perdre 3 pixels sur 4 avant de recreer 3 copies de chaque pixel, ce qui fait qu'en terme d'echantillonage on a théoriquement perdu ¾ des valeurs de gris de la composante

Avec cette opération, nous avons un gain théorique de ¾ pour les composantes bleues et rouges, soit un gain théorique total de 50% de taille mémoire.

Après le sous-echantillonage de la composante verte, on remarque que l'image reconstituée a simplement perdu en résolution par rapport à l'image originale, le processus semble totalement identique à un sous-echantillonage sur l'image couleur.

Commencons par calculer la nouvelle taille pour chaque composante :

Bleue : -echantillonage : 1/16 , quantification : 4/8 => 1/32

Rouge : -echantillonage : ¼, quantification : 5/8 => 5/32

Verte : - quantification : 6/8 => 24/32

Pour une nouvelle taille mémoire de 30/96 par rapport à l'originale soit environ 70% de gain en taille mémoire .