Quantization, Space & Discreet

Ce challenge était en fait une compression au format JPEG, découpée étape par étape avec python. Voici les étapes d'une conversion habituelle :

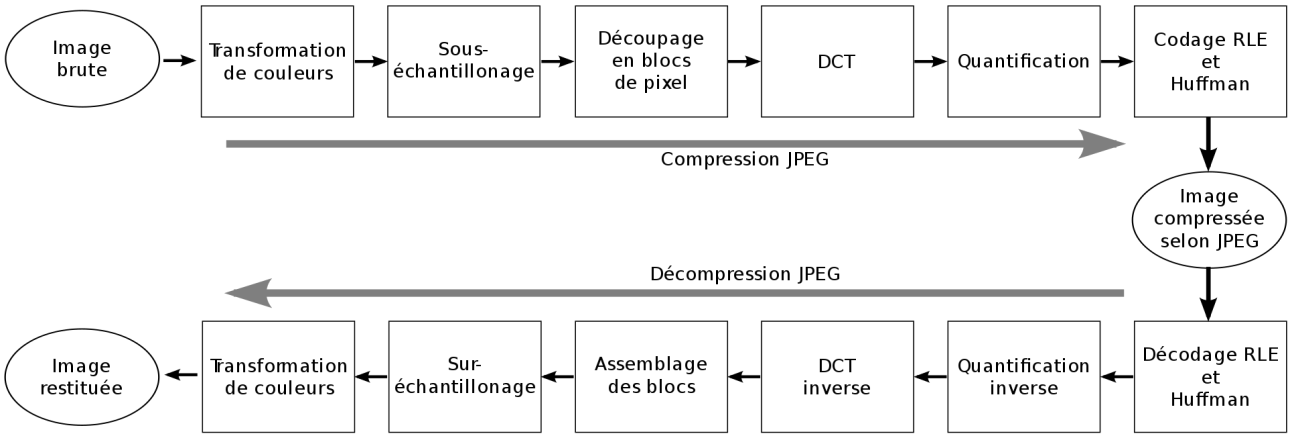
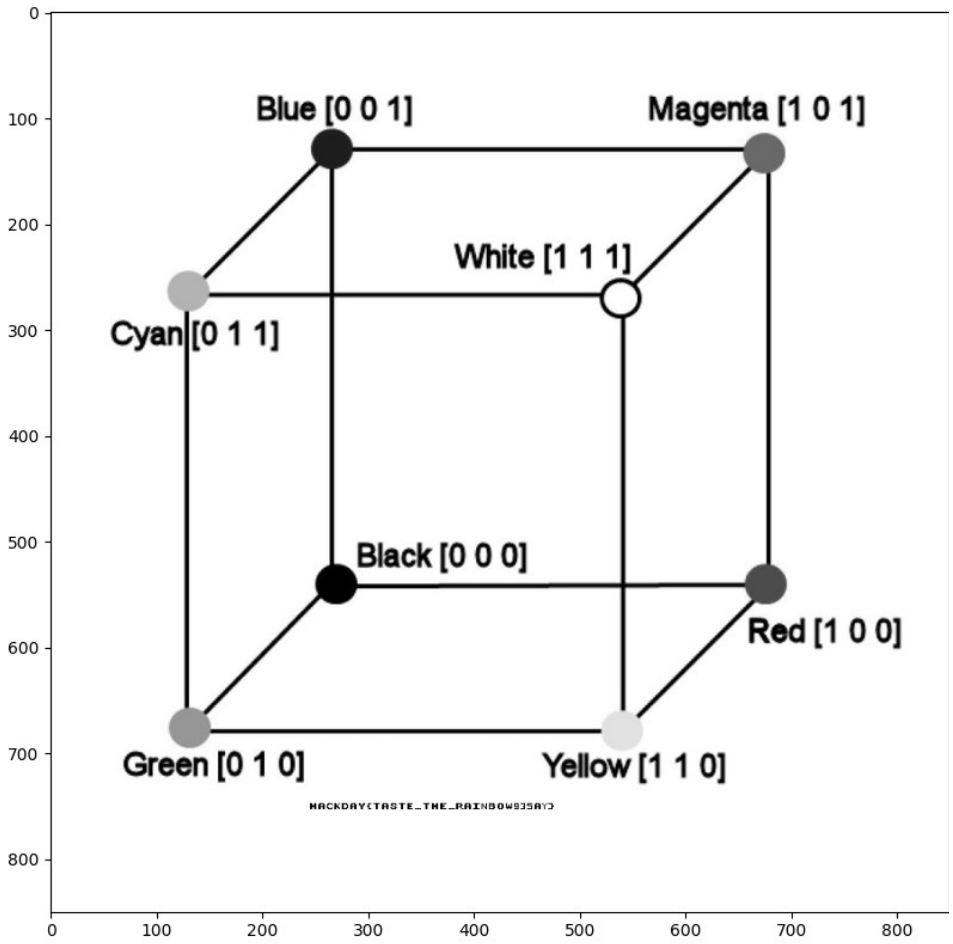


image en provenance de https://fr.wikipedia.org/wiki/JPEG

La première partie vous fournissait une image qui avait subi une transformation de couleur ainsi qu'un sous-échantillonnage sur les coefficients de couleur.

Méthode simple :

Le fait est que le coefficient de luminance Y correspond à la projection des triplets RGB sur l'axe de gris. N'étant même pas sous-échantilloné, nous pouvons directement afficher cette matrice en prenant chaque coefficient comme une quantité de gris, et nous lisons le flag sur l'image :

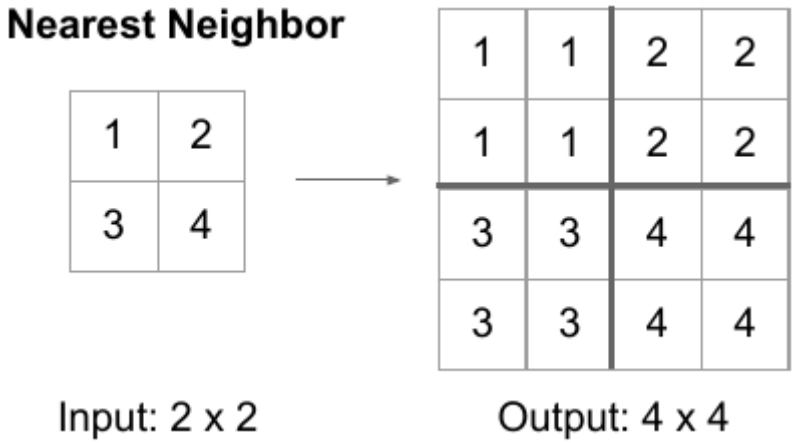


HACKDAY{TASTE_THE_RAINBOW935AY}

Méthode Propre :

En reprenant les étapes de compression dans le sens opposé, nous devons d'abord effectuer un sur-échantillonnage. En effet, l'oeil humain voit mieux les changements de luminosité que les changements de couleurs, d'où le fait que l'on se soit permis de sous-échantillonner les coefficients de couleur.

Pour sur-échantillonner, nous allons prendre chaque coefficient des matrices Cb et Cr et le dupliquer quatre fois en forme de carré :



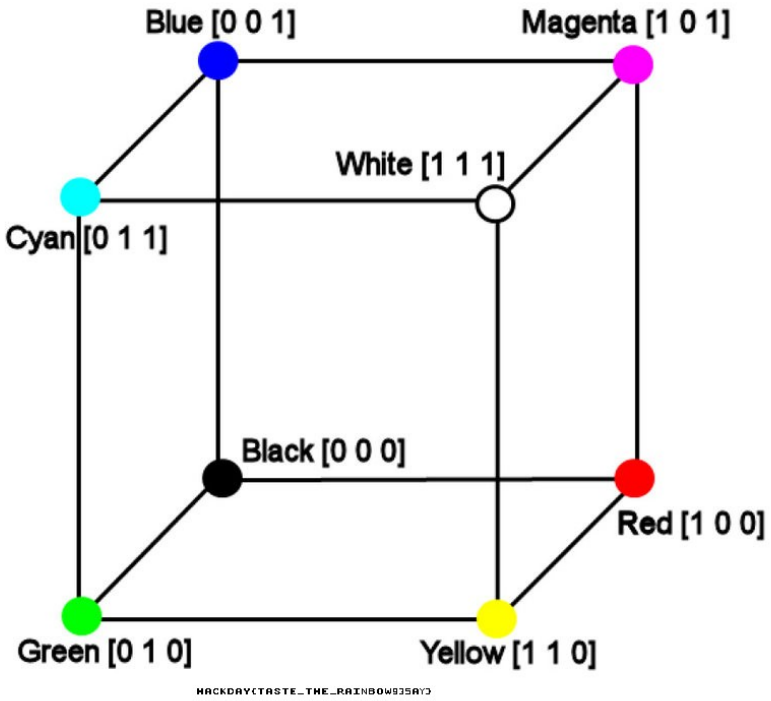
Les matrices Cb et Cr devraient maintenant avoir une taille similaire à celle de Y. Pour la suite, j'ai décidé de rassembler les trois matrices Y, Cb, Cr en une seule matrice de triplets [Y[0,0], Cb[0,0], Cr[0,0]]. Maintenant, il nous faut faire le changement d'espace colorimétrique « inverse ». Nous avons la matrice qui a servi à faire l'aller, nous n'avons donc qu'à inverser l'équation qui a servi au premier changement pour trouver l'équation à utiliser lors de changement inverse :

$$\begin{bmatrix} Y' \\ C_B \\ C_R \end{bmatrix} = \begin{bmatrix} 0.2126 & 0.7152 & 0.0722 \\ -0.1146 & -0.3854 & 0.5 \\ 0.5 & -0.4542 & -0.0458 \end{bmatrix} \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}$$

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.5748 \\ 1 & -0.1873 & -0.4681 \\ 1 & 1.8556 & 0 \end{bmatrix} \begin{bmatrix} Y' \\ C_B \\ C_R \end{bmatrix}$$

from en.wikipedia.org/wiki/YcbCr#ITU-R_BT.709_conversion

Il suffit donc d'appliquer cette formule qur chaque triplet YCbCr pour retrouver les triplets RGB tant recherchés ! Reste simplement à afficher l'image :



HACKDAY{TASTE_THE_RAINBOW@35AY}

La seconde partie incluait la séparation de chacune des matrices Y, Cb et Cr en blocs de 8*8 valeurs, suivie de la trnasformée en cosinus discret de chacun de ces blocs.

Encore une fois, il n’est pas utile de travailler sur les matrices de couleur, le flag étant écrit noir sur blanc. Nous allons donc parcourir la matrice Y bloc par bloc, et y appliquer la transformée en cosinus discret inverse (il y a une fonction toute faite *idct* dans scipy.fft). Il faut ensuite reformer une matrice bidimensionnelle en faisant un simple reshape avec la bonne taille pour chacun des axes. Après cela, il suffit d’afficher la matrice Y obtenue, similairement à ce qui avait été fait dans la première partie ^^

mmmmm, monke

Si l’on souhaitait obtenir l’image avec les couleurs (dans son intégrité), il fallait seulement faire les mêmes calculs sur Cb et Cr que ceux faits sur Y (IDCT et regroupement des blocs), et poursuivre avec la méthode de la première partie. On obtient donc :

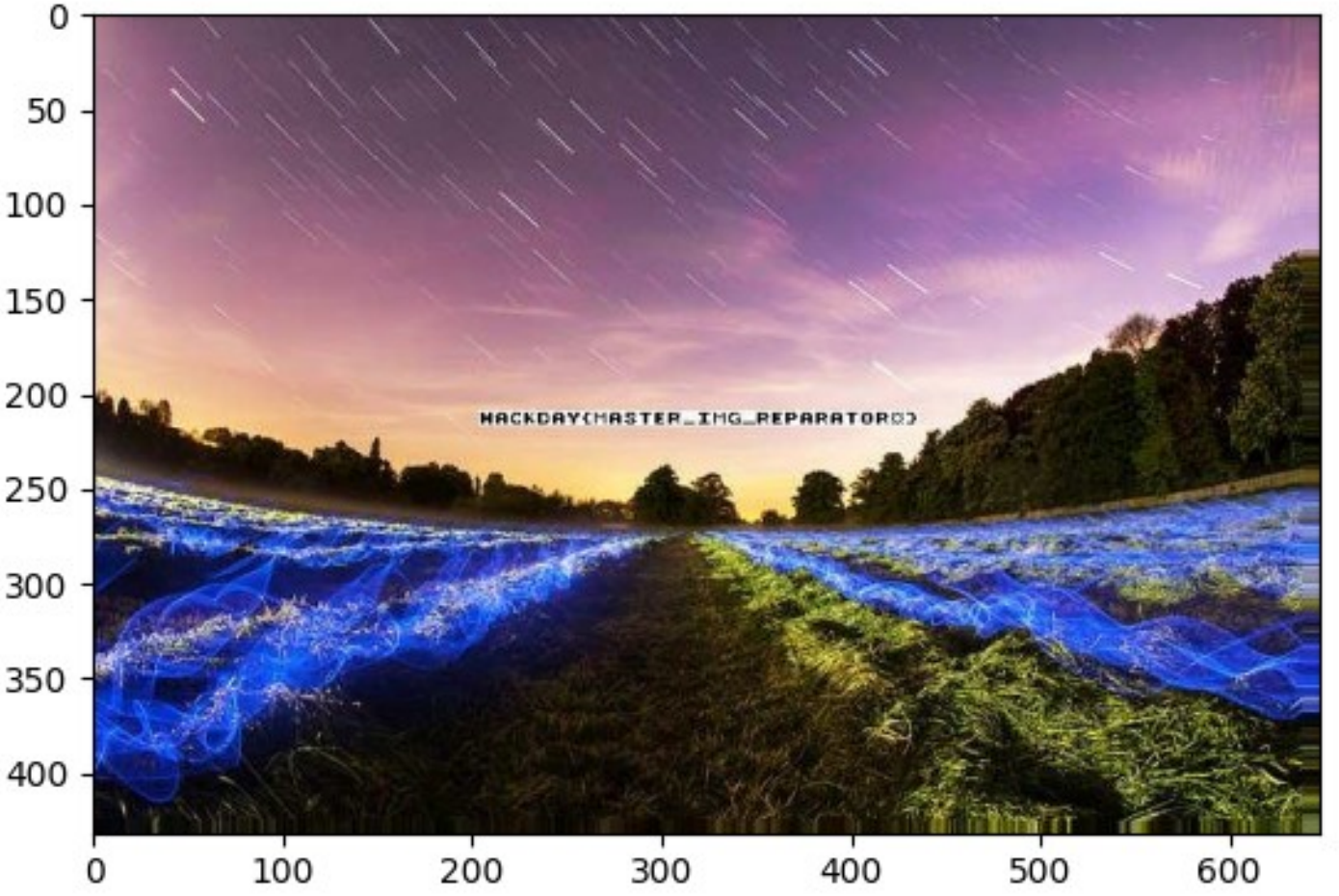
HACKDAY{AS_DISCRET_AS_NINJAS!!!!!!!!}

La dernière partie consistait à ajouter l'étape de la quantification. Elle sert à ôter les hautes fréquences des signaux qui sortent de la transformée en cosinus discret. En effet, les images très continues auront de faibles coefficients en bas à droite de leurs matrices 8*8. Nous allons donc diviser toutes ces valeurs pour en approcher le maximum de zéro car zéro se stocke bien, surtout s'il y en a plusieurs à la suite. Nous avons donc donné la matrice de quantification qui a été utilisée :

$$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}.$$

Il faut donc simplement prendre chaque bloc 8*8 et faire un produit de Hadamard avec la matrice de quantification donnée. Après ce produit, il suffit simplement de repasser par toutes les étapes d'avant et on obtient :

Et en le faisant proprement :



HACKDAY{MASTER_IMG_REPARATOR¤}

Chelinka Latov, pour le HACKDAY du 14 au 15 mai