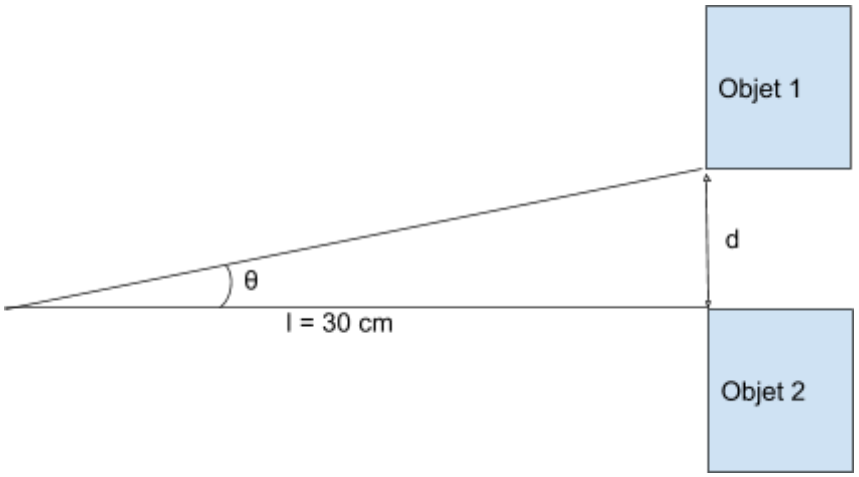
Traitement d'image

Rapport

TP 1

Définition

Pour qu'un objet soit distinguables par l'oeil, il faut que la distance entre deux objets soit d'au moins $d = 30 \times tan\theta = 0.0089\,cm = 89\,\mu m$ (cf. schéma ci-dessous).



Un pixel d'un smartphone de 5 pouces avec une qualité HD 720 est de l'ordre de : $6.2 / 720 = 0.0086\,cm = 86\,\mu m$. Ainsi deux pixels sont indiscernables l'un de l'autre si le smartphone est tenu à 30 cm de l'oeil (d'après le résultat calculé à la réponse précédente), donc la définition est suffisante.

Pour un écran 1080p de 22 pouces, la taille d'un pixel est de :
$d = 27.4 / 1080 = 0.025\,cm = 25\,\mu m$
il faut donc être à une distance $l < d / tan(0.017°) = 84.25\,cm$ pour pouvoir distinguer les pixels.

Pour une personne se trouvant à $l = 2m$ de son canapé, pour arriver à distinguer les pixels, il faut que sa taille pixel soit de $d = 0.59\,mm$ donc pour un écran HD1080 l'écran devrait être de hauteur : $h = d * 1080 = 637\,mm = 63.7\,cm$ et donc de diagonale d'environs 55 pouces. Cette personne ne verra la différence entre les 2 résolutions uniquement si son écran fait au moins cette taille.

Lecture de l'image *mandrill.bmp*

```
>> imdata = imread('mandrill.bmp');
>> size(imdata)

ans =
   512   512 3
```

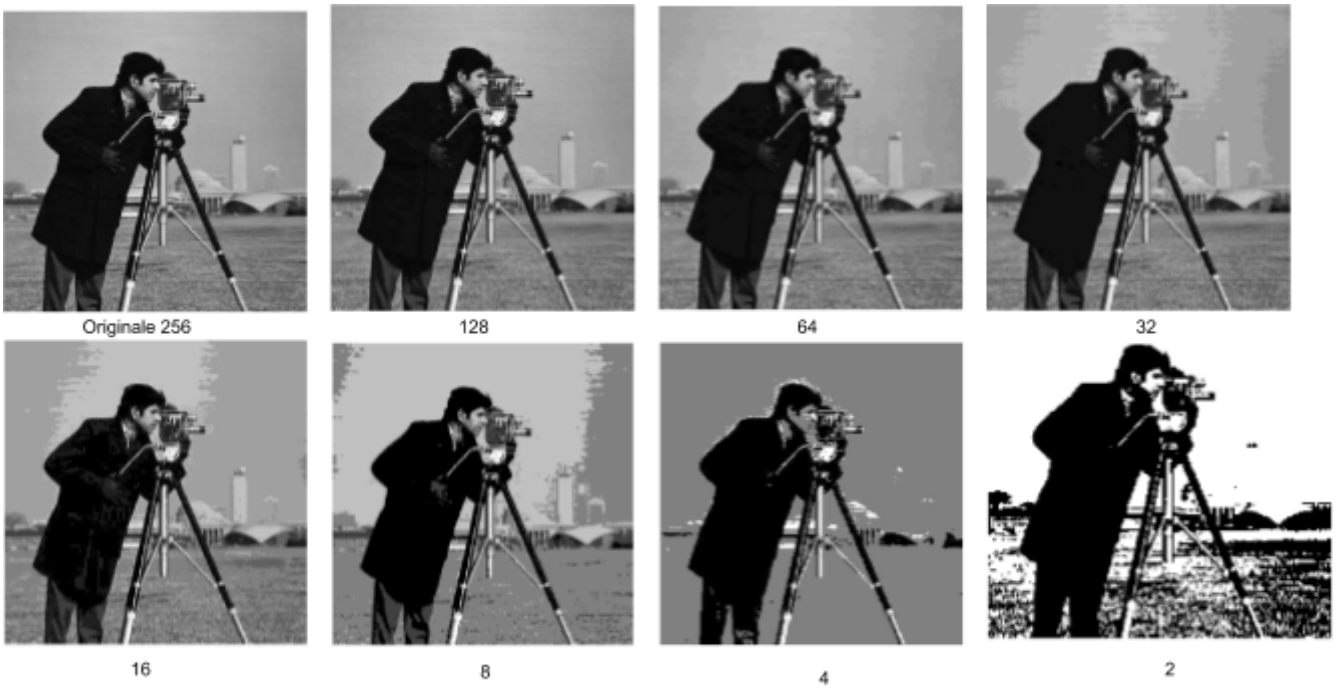
Le format bmp sauvegarde les images sous forme de 3 matrices (une pour le rouge, une pour le vert et une pour le bleu) de taille $n \times m$. Ainsi cette image est de définition 512 x 512.

La taille théorique sur le disque de cette image est donc $s = 512 \times 512 \times 3 = 786\,432\ octets$, en effet chaque valeur des matrice est codée sur un octet. Toutefois, la taille réelle de cette image sur le disque est de $786\,486\ octets$, il y a donc 54 octets utilisés pour les métadonnées du fichier (la commande $imfinfo('mandrill.bmp')$ permet de trouver celles-ci sous l'appellation *ImageDataOffset*).

Quantification

```
        Filename: '\\teraetu.univ-lyon1.fr\homeetu\p1402690\My Documents\Traitement_image\cameraman.jpg'
     FileModDate: '17-Nov-2019 22:33:08'
        FileSize: 10515
          Format: 'jpg'
   FormatVersion: ''
           Width: 256
          Height: 256
        BitDepth: 8
       ColorType: 'grayscale'
 FormatSignature: ''
 NumberOfSamples: 1
    CodingMethod: 'Huffman'
  CodingProcess: 'Sequential'
         Comment: {}
```

Le niveau de gris est donc codé sur 8 bits.
Voici les images obtenues en utilisants les différents nombres niveaux de gris :



Echantillonnage

Image sous-échantillonnée 2 fois :

Sur-échantillonnée

Interpolée

4 fois :

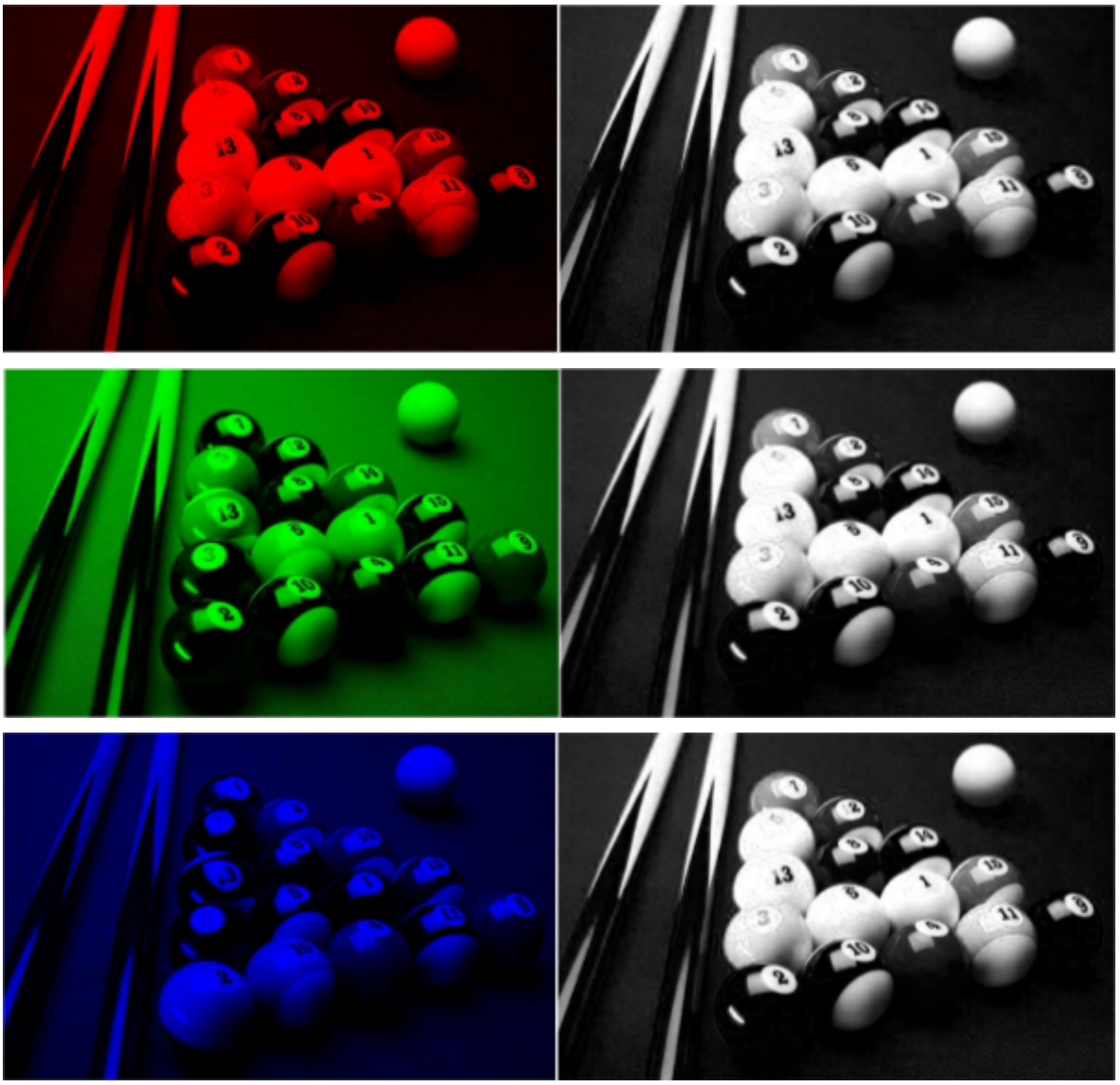
Sur-échantillonnée

Interpolée

On constate que les images, bien qu'ayant été sur-échantillonnées pour retrouver leur taille d'origine, ont perdu en quantité d'information. En effet, le sous-échantillonnage implique une perte d'information (on perds la moitié ou les ¾ des pixels), et cette information ne peut pas être retrouvée par un simple sur-échantillonnage. Cependant, en ajoutant une interpolation, on peut "lisser" notre image pour la rendre moins "pixelisée", mais cela implique quand même une perte d'information.

Espaces colorimétriques

RGB :

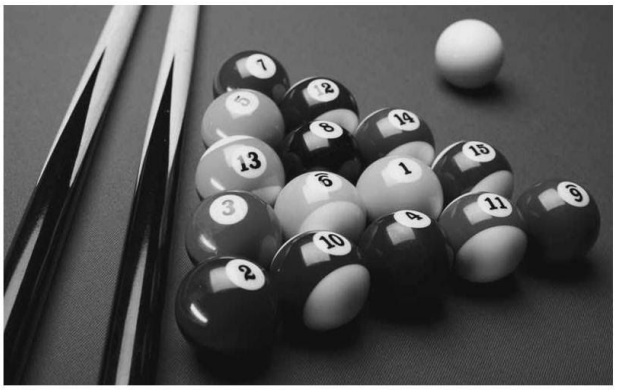


Pour les canaux RGB, les plages de valeurs de chaque couleur sont les plages de valeurs maximales, soit [0 ; 255].

YUV:

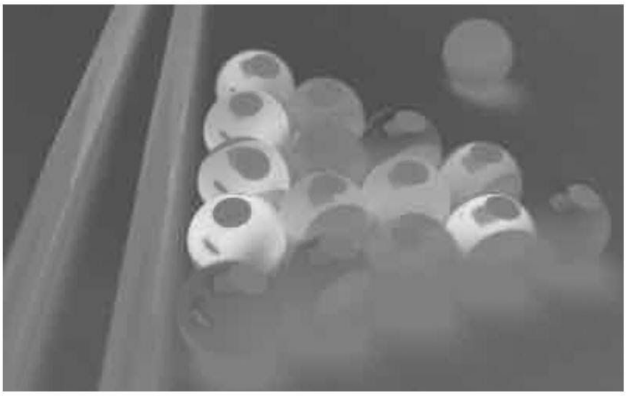
Les canaux sont affichés en nuances de gris. (Le code se trouve dans tp1.mlx)

Canal Y :



Canal U :

Canal V :

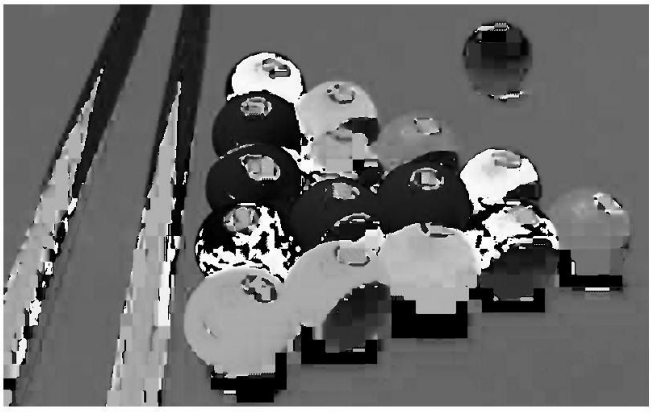


Les valeurs sont également comprises entre 0 et 255 en valeurs entières..

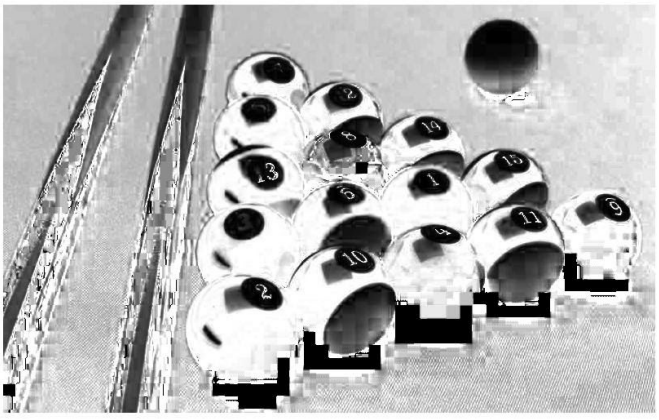
HSV:

Les canaux sont affichés en nuances de gris. (Le code se trouve dans tp1.mlx)

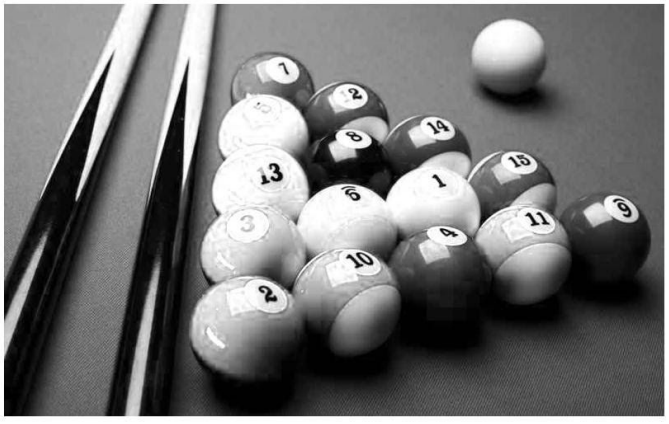
Canal H:



Canal S:



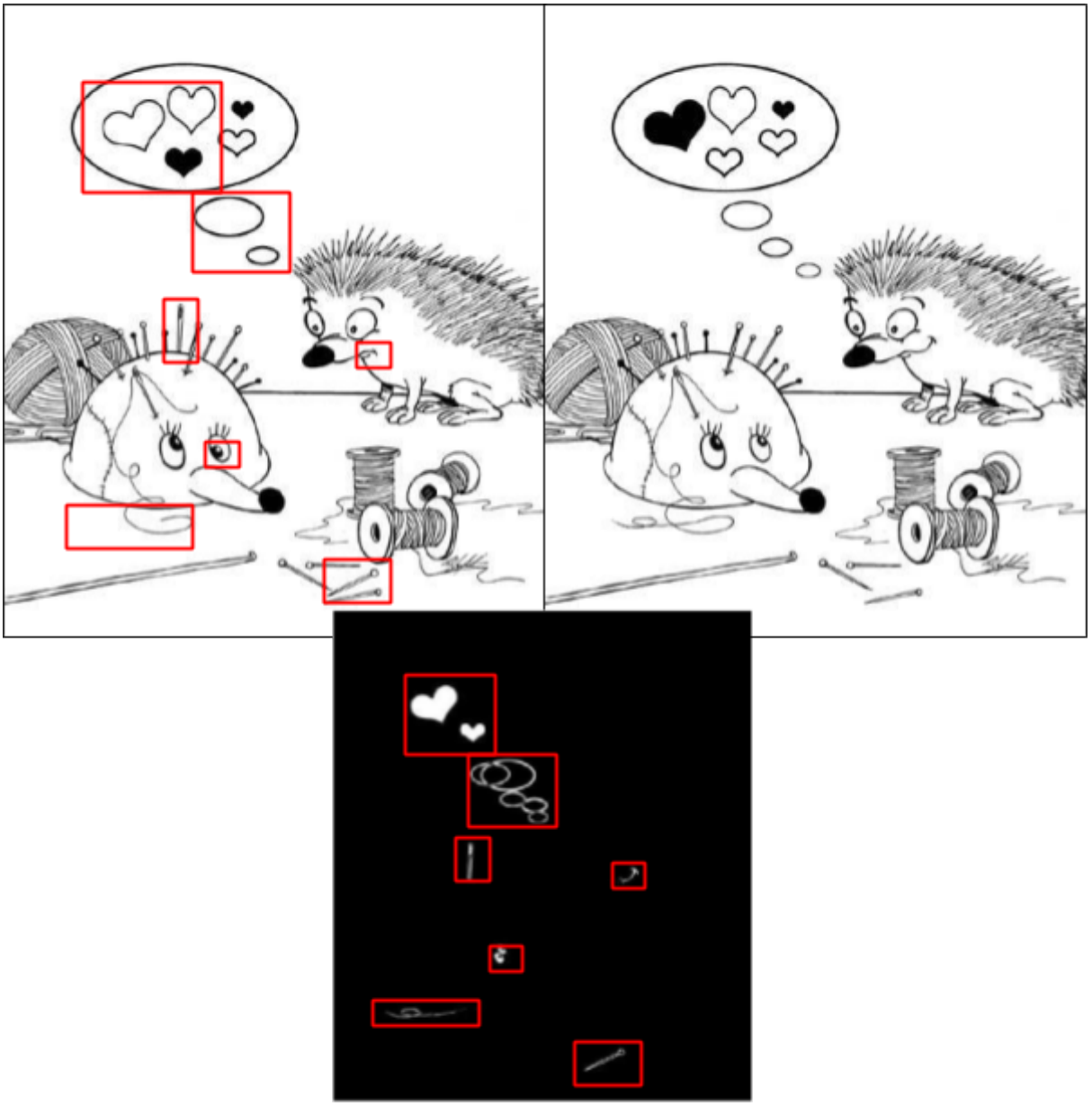
Canal V:



Les valeurs sont cettes fois comprises entre 0 et 1 (en nombres décimaux).

7 différences

Pour trouver les différences il suffit de soustraire les deux images. Plus précisément, nous avons calculé les deux soustractions (img1-img2 et img2-img1, en valeur absolue), et nous avons additionné les résultats. Cela nous donne une image où tous les pixels différents d'une image à l'autre sont affichés en blanc.



TP2

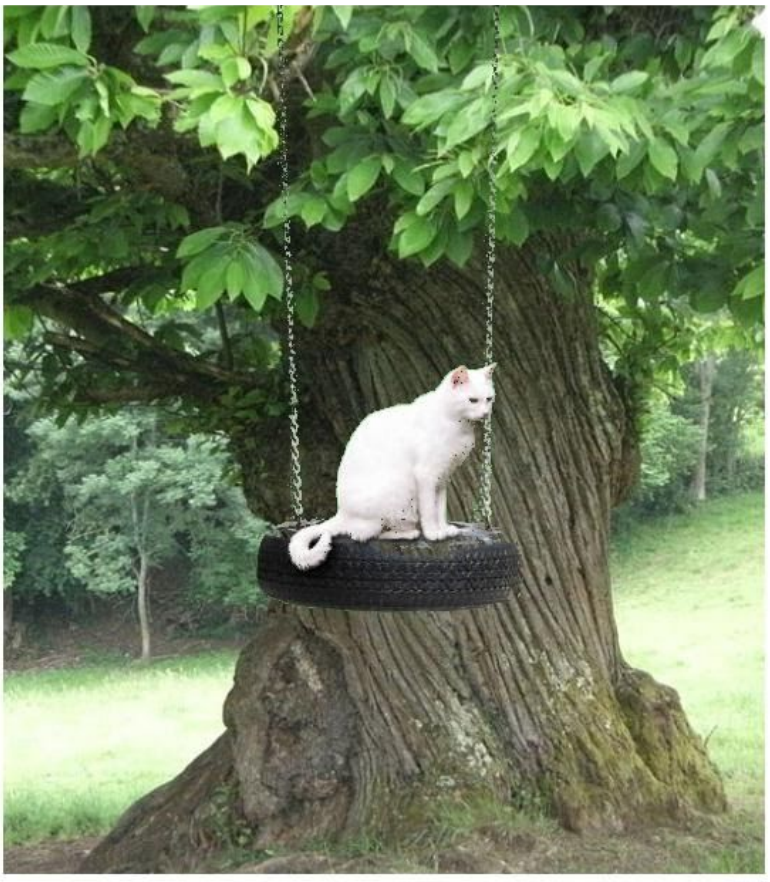
Arbre à chat

Afin de remplacer le fond vert, nous créons un masque (de la taille des images) qui a pour valeure ‘vrai’ si le pixel correspondant est de la même couleur que le fond. Pour ce faire, nous prenons comme référence le premier pixel pour chaque channel :

```
fond_r = chat(1,1,1);
fond_g = chat(1,1,2);
fond_b = chat(1,1,3);
```

Ce qui nous donne la valeur du pixel ‘fond vert’ pour chacun des channels. Ainsi notre masque sera ‘vrai’ si la valeur du pixel sur l’image de chat n’est pas celle du pixel ‘fond vert’. En pratique nous avons ajouté une marge, ce qui permet de supprimer les pixels qui ont une valeur ‘proche’ du pixel ‘fond vert’ afin d’avoir un meilleur résultat visuellement (nous avons choisi ces valeures empiriquement).

Enfin, nous appliquons ce masque sur les images : si ‘vrai’ on utilise le pixel de l’image de chat, sinon, celui de l’image de l’arbre. Voici le résultat obtenu :



Le fait que cette image est un montage reste assez visible, mais de loin on pourrait y croire.

Nous n’avons pas réussi à obtenir de meilleur résultat visuellement, même en utilisant des marges ‘asymétriques’ pour certains channels.

Segmentation

Voici les résultats obtenus :

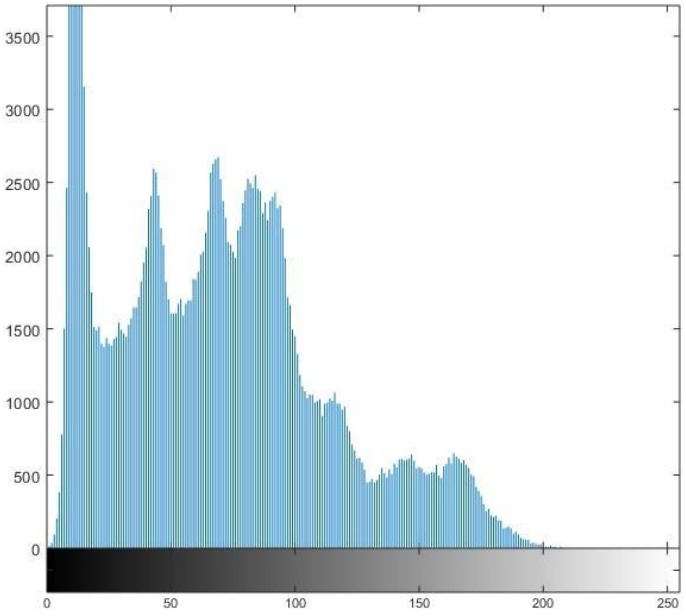
Image initiale | Résultats seuil “à la main” | Résultats seuil “*graythresh*”

Afin de trouver un seuil “à la main” nous avons fait plusieurs test pour trouver un résultat convenable (seuil = 0.7), la fonction graythresh nous donne directement un seuil, ici égal à 0.6275.

Dans notre image nous avons plus de “faux positifs” (des objets n’étants pas des noyaux) qui apparaissent, mais nous capturons plus les noyaux que le seuil *graythresh* qui laisse apparaître des trous dans les noyaux. La meilleure solution dépend de l’utilisation que nous voudrions faire d’une telle segmentation.

Dynamique

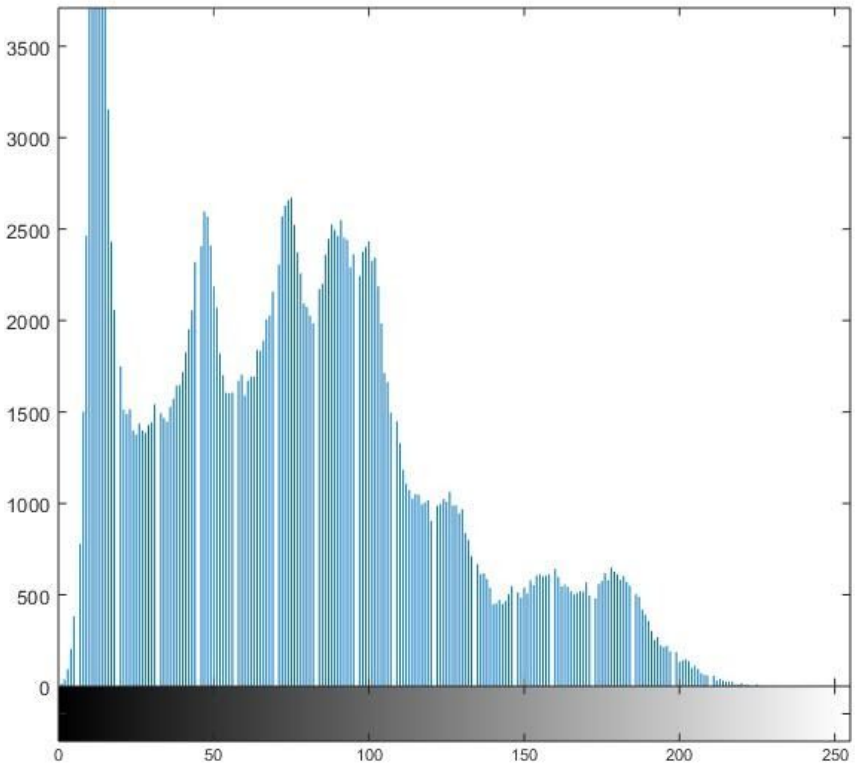
En appliquant la formule du cours, on trouve C = 1 (max = 235, min = 0)



Ci-contre, l’histogramme de l’image Lena.

On constate que les valeures des pixels ne sont pas réparties sur tout le spectre (max = 235).

En appliquant la formule du cours (ce qui revient à multiplier l'image par 255/235), nous obtenons alors l'histogramme suivant :



Cette fois-ci le maximum est bien à 255 au lieu de 235.
Visuellement, on a :

Image originale | Image "étalée" | Différence

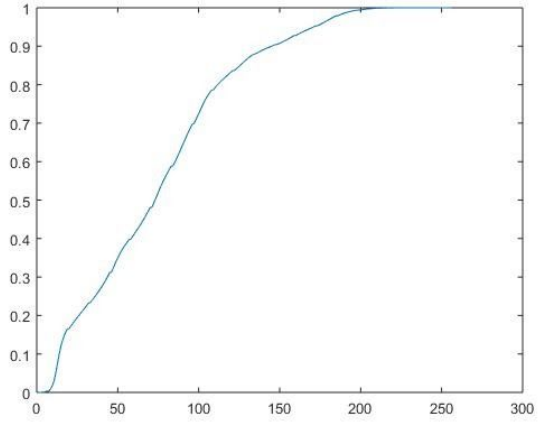
Lorsqu'on compare les deux images, la différence n'est pas flagrante, mais si l'on soustrait les deux images, on voit bien apparaître une petite différence.

Egalisation

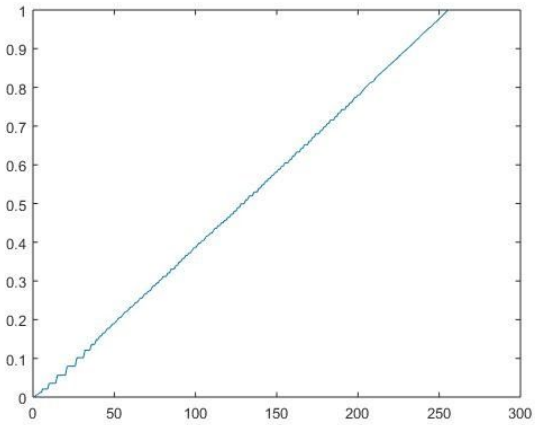
Voici les résultats obtenus :

image originale

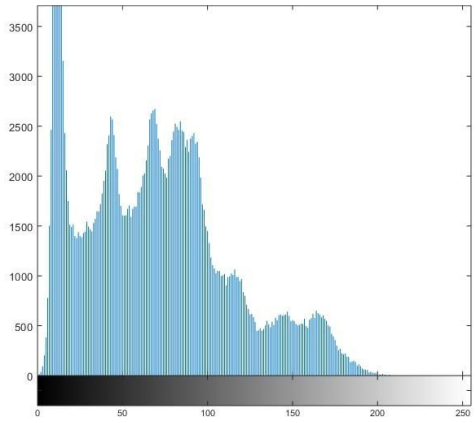
image égalisée



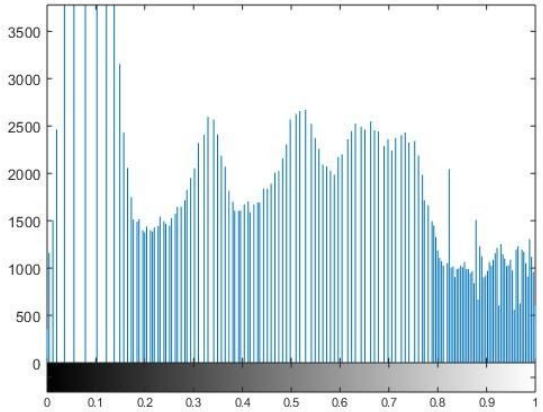
Histogramme cumulé image originale



Histogramme cumulé image égalisée



Histogramme image originale



Histogramme image égalisée

On constate une claire différence visuelle. En regardant l'histogramme cumulé, on peut confirmer que l'image est bien égalisée.

`Mandrill.bmp :`

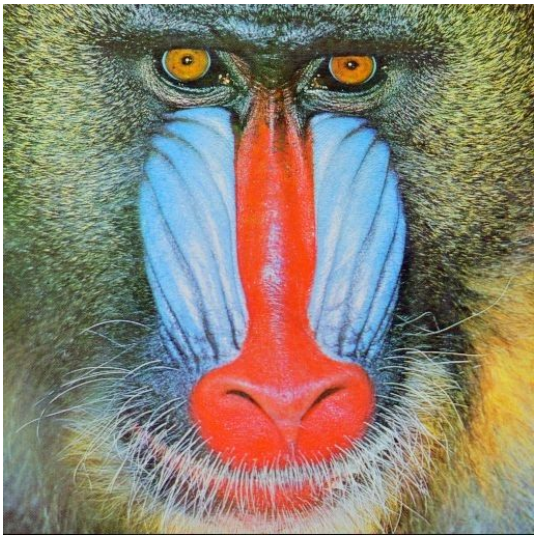


Image originale

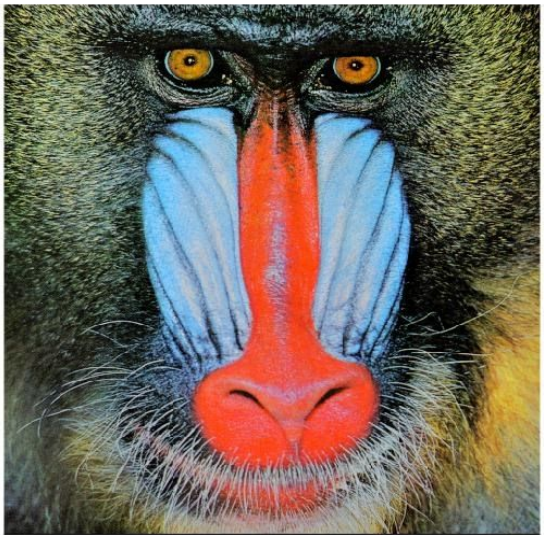


Image égalisée

Pour obtenir ce résultat, nous avons convertit l'image en HSV, puis nous avons appliqué l'égalisation sur le channel V de l'image originale en utilisant la même méthode que pour Lena_nb.

TP3