Dimitri Charneux

Rendu tp:

Modification d'images couleur.

Table des matières

Introduction

Pour représenter une image, il existe différent format comme le format le plus connue, le RGB, ou des formats ayant d'autres caractéristiques comme l'HSB.

Durant ce TP, nous allons utiliser ces différents format pour manipuler au mieux des images est détecter quels changement ont été appliqués à quelles images.

1. Manipulation de la luminance

Nous avons tout d'abord manipulé la luminance des images pour retrouver avec quelle modification de luminance nous pouvons passer de certaines images à d'autres.

Pour ce faire, nous avons utilisé le plugin « Color Inspector 3D » pour modifier la luminance en faisant varier un slider.

Par exemple, pour les deux images suivantes, on constate une différence de luminosité entre les deux images.

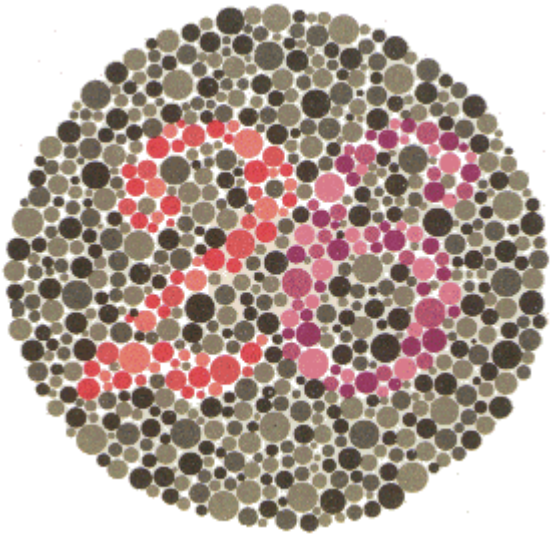


Illustration 1: cas_1_luminance.

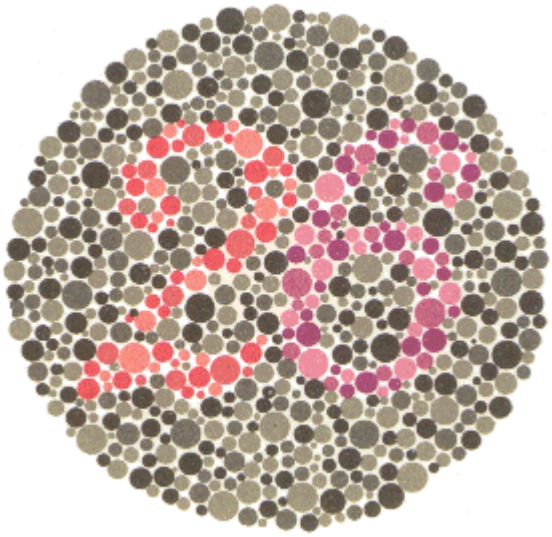


Illustration 2: cas_1_dalton_26.

Nous allons confirmer cette différence grâce au plugin cité précédemment en modifiant la luminance de l'image de droite par -20.

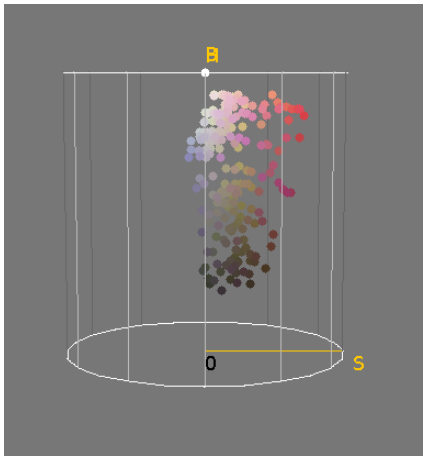


Illustration 3: cas_1_luminance_HSB.

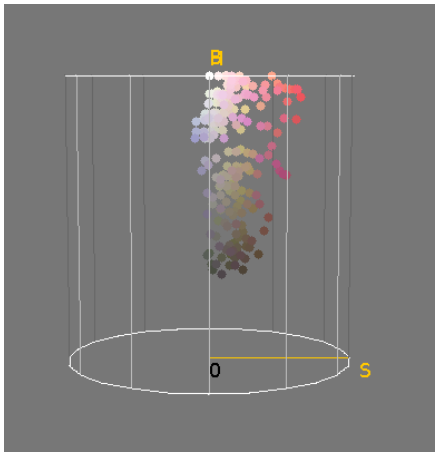


Illustration 4: cas1_dalton26_HSB.

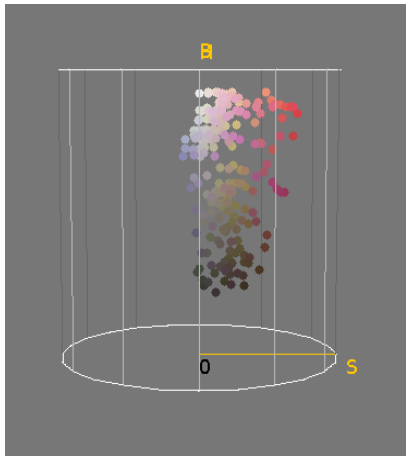


Illustration 5: cas_1_dalton_26 avec luminance -20.

On voit bien que l'illustration 3 et l'illustration 5 sont presque identique tandis que sur l'illustration 4, les points sont tous plus haut que sur les autres illustrations.

Nous avons ensuite réalisé la même opération sur les 2 autres paires d'images ci-dessous. Pour le cas n°2, nous avons enlevé 80 de luminance à l'image dalton pour obtenir l'image originale. Pour le cas n°3, nous avons enlevé 30 de luminance à l'image dalton pour obtenir l'image originale.

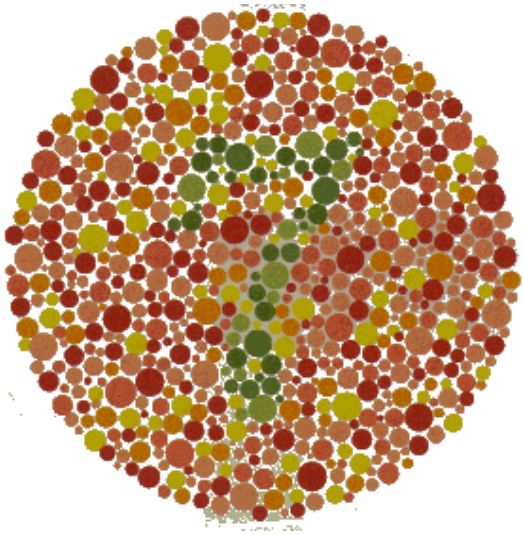


Illustration 6: cas_2_luminance.

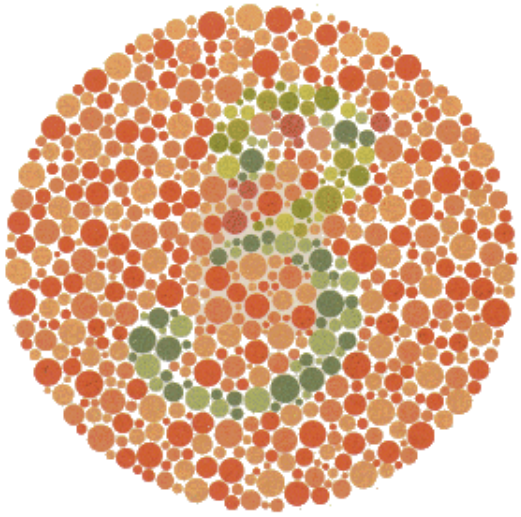


Illustration 7: cas_3_luminance.

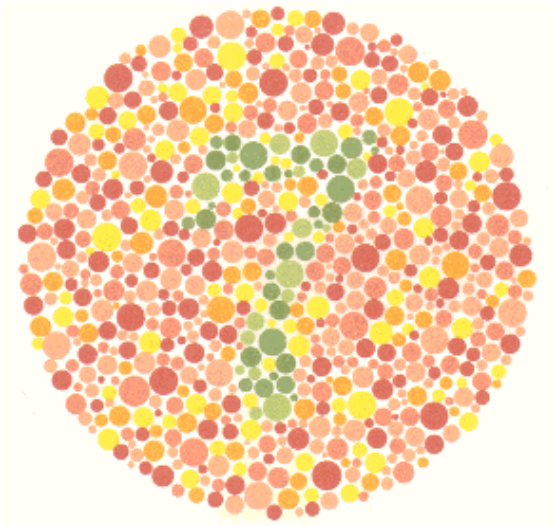


Illustration 8: cas_2_dalton7.

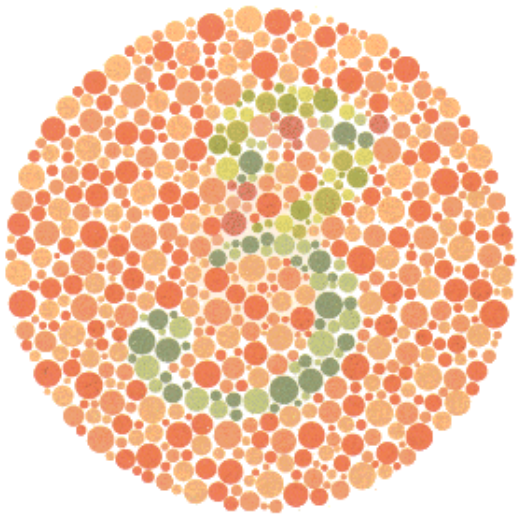


Illustration 9: cas_3_dalton3.

Nous avons ensuite réalisé une macro permettant de modifier la luminance sans passer par le plugin 3D inspector. Cette macro, présente en annexe, va demander à l'utilisateur une valeur de luminance à ajouter et va ajouter cette valeur à chacune des couleurs pour chaque points de l'image.

Avec cette macro, nous obtenons les mêmes valeurs qu'avec le plugin, c'est à dire 20 pour le cas 1, 80 pour le cas 2 et 30 pour le cas 3 pour passer de l'image luminance à l'image dalton.

Pour vérifier nos résultats, nous avons effectuer la différence entre l'image dalton et l'image luminance modifiée par la macro. Pour ce faire, nous avons utilisé l'outil Process-Image Calculator.

Pour les cas 1 et 3, nous retrouvons exactement la même image. En effet, la différences entre les images luminance et dalton sont des images toutes noires comme le montre l'image et l'histogramme représentant cette différence. Nous obtenons exactement le même résultat pour la cas 1 et 3.

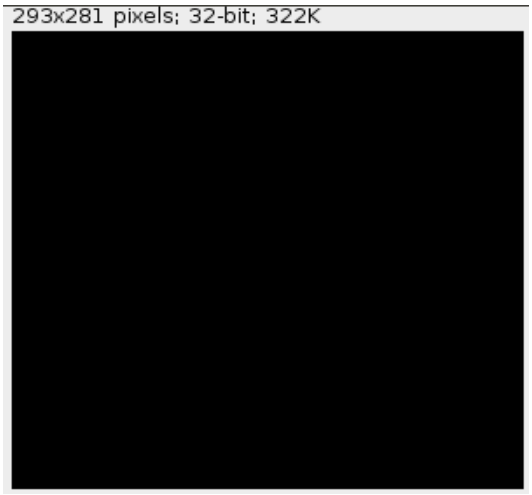


Illustration 10: Différence entre l'image dalton et l'image luminance modifiée.

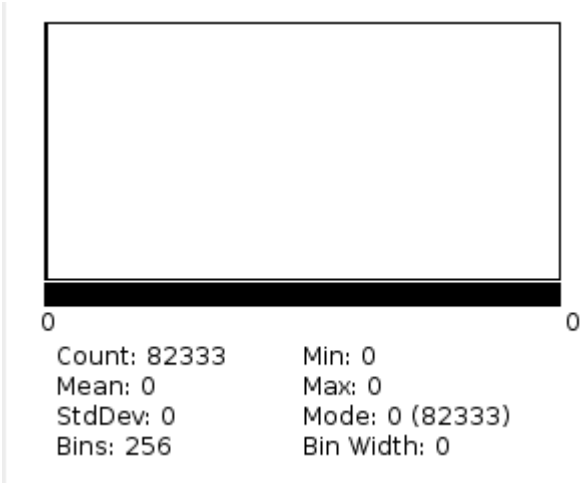


Illustration 11: Histogramme de cette différence.

Le cas 2 nous donne un résultat différent. En décochant le résultat en image 32 bits, nous obtenons le même résultat que pour les deux autres cas. Par contre, en activant le résultat en 32 bits, nous obtenons l'image et l'histogramme suivant :

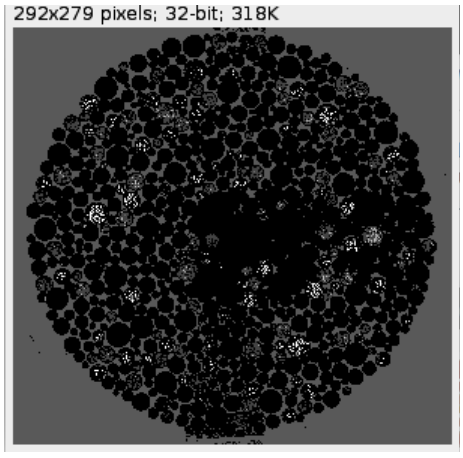


Illustration 12: Différence entre l'image dalton et l'image luminance modifiée du cas 2.

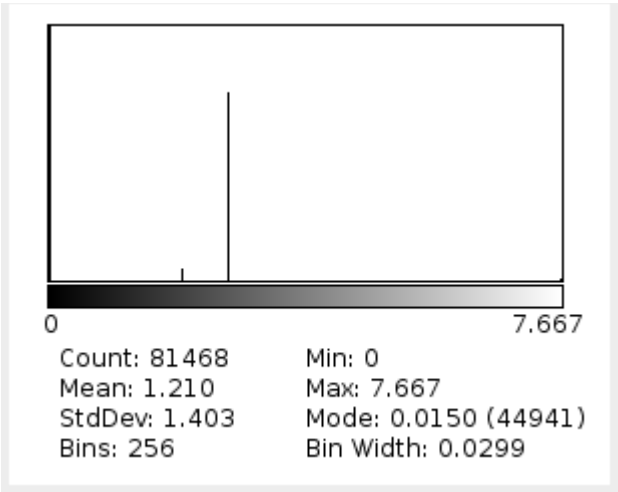


Illustration 13: Histogramme de cette différence.

Nous constatons grâce à cette image, que le fond de l’image dalton et celui de l’image luminance sont différents. En regardant de plus prêt, on peut voir que le fond de l’image dalton 7 est rosée tandis que celui de l’image modifié est parfaitement blanc. Nous pouvons voir également quelques points blanc qui montre que la modification de la luminance ne transforme pas parfaitement l’image.

Maintenant que nous avons vu comment manipuler la luminance d’une image, nous allons voir comment manipuler sa saturation.

2. Rétablissement de la saturation

Dans cette partie, nous allons voir comment la saturation permet de modifier des images. Pour cela, nous allons utiliser les deux paires d'images suivantes.

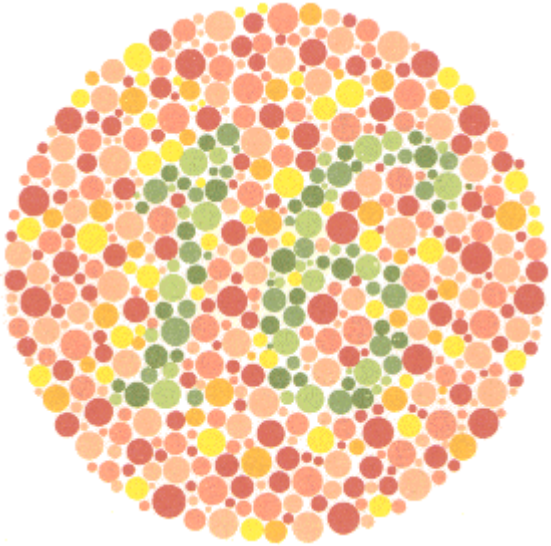


Illustration 14: Cas 2, image originale.

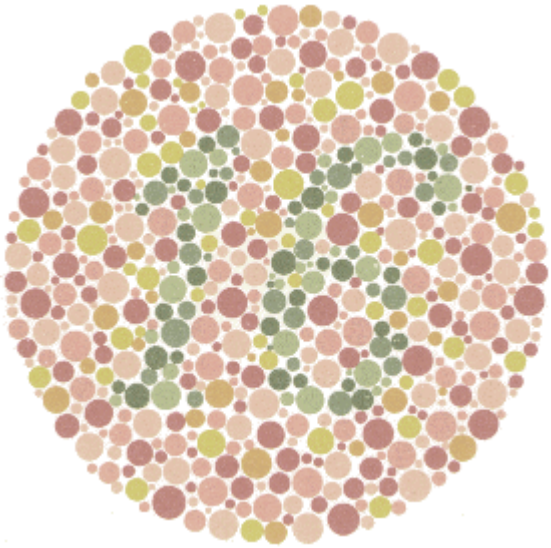


Illustration 15: Cas 2, image modifiée.

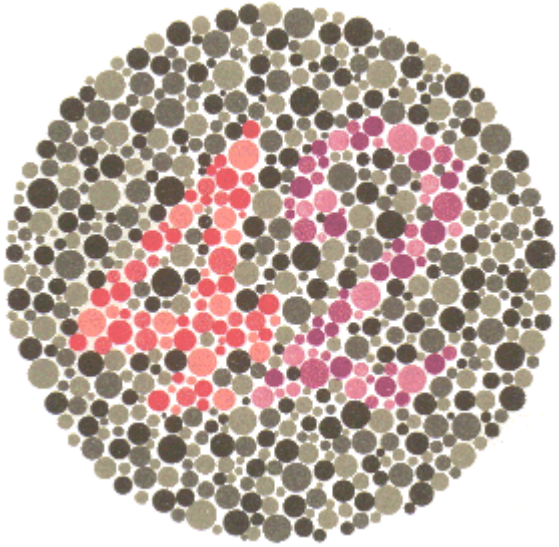


Illustration 16: Cas 1, image originale.

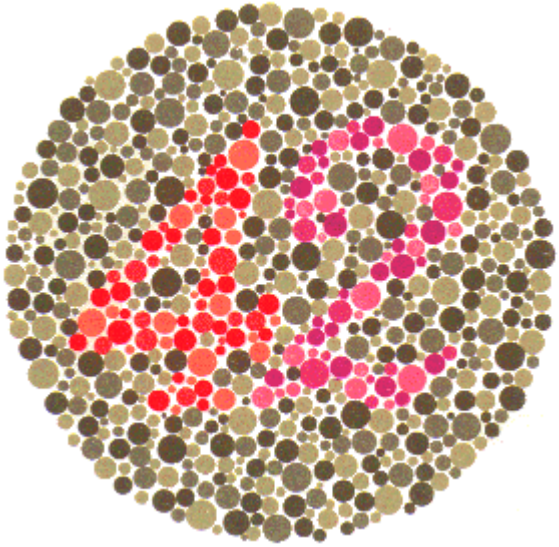


Illustration 17: Cas 1, image modifiée.

Pour passer des images modifiées aux images originales, nous avons utilisé le plugin « Color Inspector 3D » pour modifier la saturation des images. Pour l'image du cas 2, nous avons multiplié la saturation par 2,04 et pour le cas 1, nous l'avons multiplié par 0,55.

Nous avons ensuite réalisé une macro, présente en annexe, permettant de réaliser cette modification directement.

Dans cette macro, nous commençons par demander à l'utilisateur la multiplication qu'il veut effectuer à la saturation. Nous allons ensuite convertir l'image de RGB à HSB puis la séparer en trois images, une image représentant la luminance, une représentant la saturation et une représentant

la teinte. Voici les lignes de code effectuant ces opérations :

```
run("Color Space Converter","from=RGB to=HSB white=D65");
run("Split Channels");
```

Nous avons ensuite multiplié la saturation par le nombre entré par l'utilisateur puis reconstituée l'image originale en réassemblant les 3 images séparées précédemment. Il ne reste plus qu'à refaire la transformation de HSB à RGB. Les lignes de code suivantes ont été utilisé pour effectuer ces opérations :

```
selectWindow(titre + " (HSB) (green)");
run("Multiply...", "value=" + valeur);

run("Merge Channels...","c1=["+titre+" (HSB) (red)] c2=["+titre+" (HSB)
(green)] c3=["+titre+" (HSB) (blue)] ignore");
run("Color Space Converter","from=HSB to=RGB white=D65");
```

C'est tout pour la saturation, nous allons maintenant passer à la transformation de la teinte.

3. Transformation de la teinte

Pour cette partie, nous avons dû modifier la teinte de l'image pour transformer l'image bleu en l'image rouge ci-dessous.

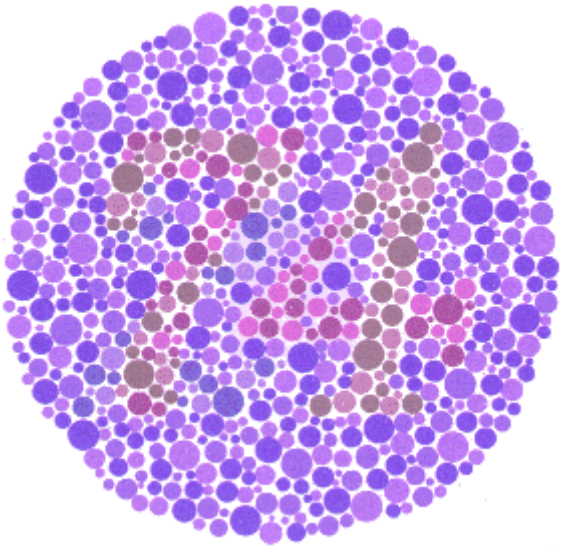


Illustration 18: Image à modifier.

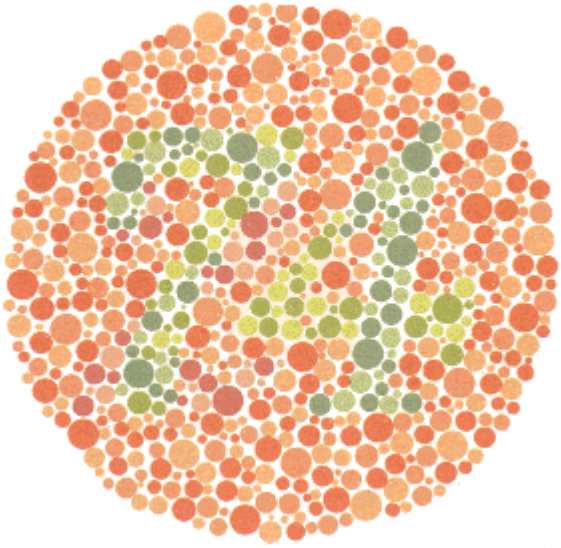


Illustration 19: Image originale.

La macro utilisée pour transformer cette image est disponible en annexe. Cette macro ressemble fortement à la précédente. Nous commençons par demander une valeur à l'utilisateur, puis nous séparons l'image en trois images représentant ses caractéristiques HSB. Ensuite, le traitement est différent, nous allons modifier la valeur entrée par l'utilisateur puis l'ajouter à la teinte de l'image grâce aux code suivant :

```
valeurRed = (valeurRed * 256 )/360;
selectWindow(titre + " (HSB) (red)");
run("Add...", "value=" + valeurRed);
```

Nous allons ensuite, comme pour l'autre macro, réassembler les trois sous image puis reconvertir le tout en RGB.

La transformation de la valeur saisie par l'utilisateur permet de retranscrire la valeur entrée par l'utilisateur dans le cercle des couleurs affiché en annexe.

Pour transformer l'image de gauche en l'image de droite, il faut donner comme valeur -240 à la macro.

Maintenant que nous avons vu comment modifier la teinte d'une image, nous allons analyser la différence entre deux images dans un espace de couleur adapté.

4. Analyse dans des espaces de couleur adaptés

Dans cette partie, nous allons voir la différence entre les deux images suivantes.

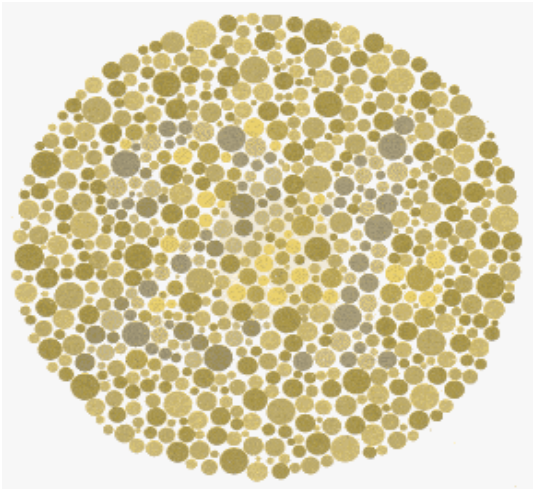


Illustration 20: Cas 3, dalton 74 modifié pour l'exercice.

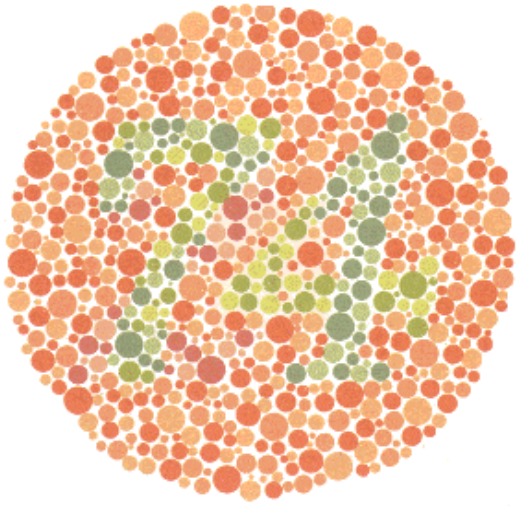


Illustration 21: Cas 3, dalton 74.

Pour identifier la différence entre les deux images, nous avons utilisé le plugin « color inspector 3d » en mode HSB. Nous obtenons les images suivantes :

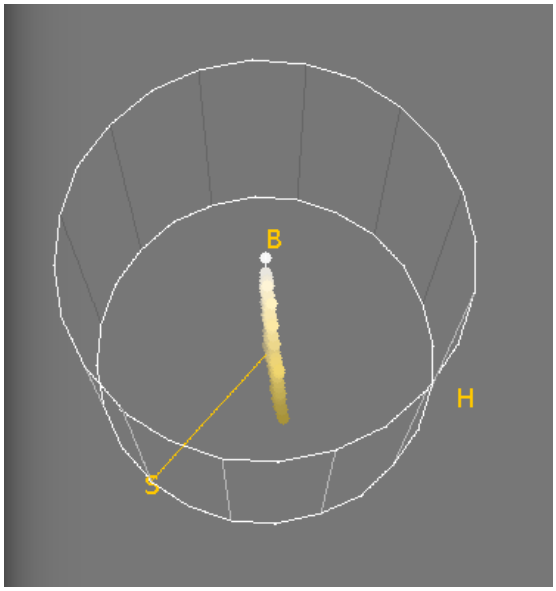


Illustration 22: Cas 3, dalton 74 modifié en HSB.

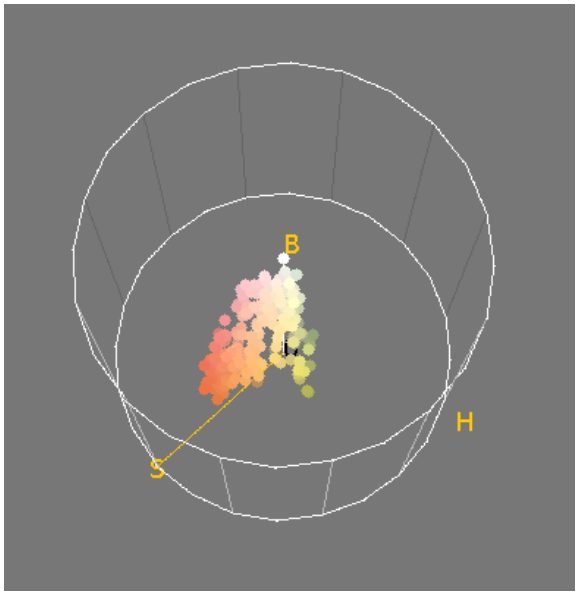


Illustration 23: Cas 3, dalton 74 en HSB.

Grâce à ces deux images au format HSB, on peut voir que pour l'image modifiée, toutes les couleurs ont été mise dans la même teinte, alors que pour l'image originale, les couleurs appartiennent à différentes teintes. On voit bien que dans l'image modifiée, le nombre est impossible à voir car tout les points appartiennent à la même teintes de couleur, tandis que dans l'image originale, le nombre a une teinte verdâtre alors que le fond a une teinte rougeâtre.

Nous allons maintenant voir la différence entre deux autres images.

5. Modification de la luminance adaptée

Dans cette partie, nous allons utilisé le même procéder que dans la partie précédente pour trouver la différence entre deux images. Voici les deux images que nous allons analyser ainsi que leur représentation HSB.

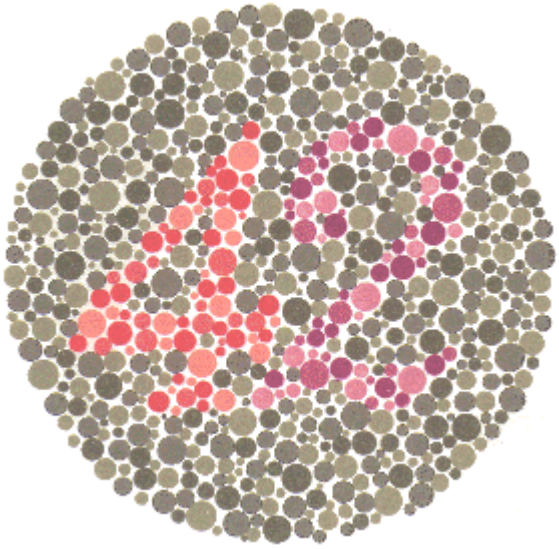


Illustration 24: Cas 1, image modifié pour l'exercice.

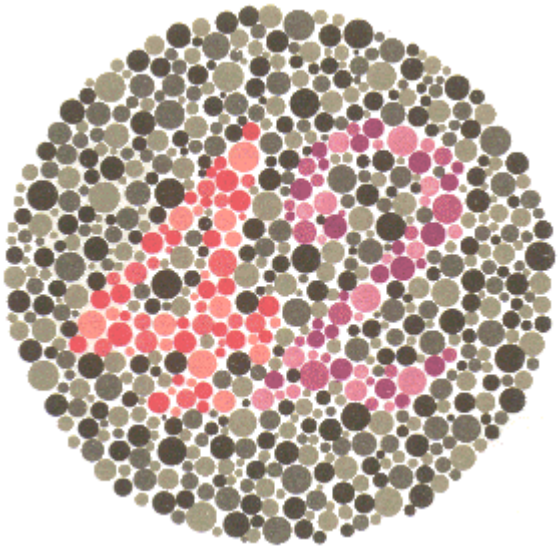


Illustration 25: Cas 1, dalton 42.

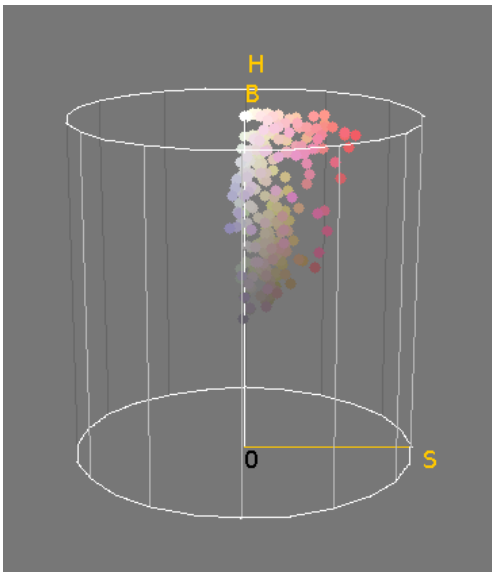


Illustration 26: Cas 1, image modifié en HSB.

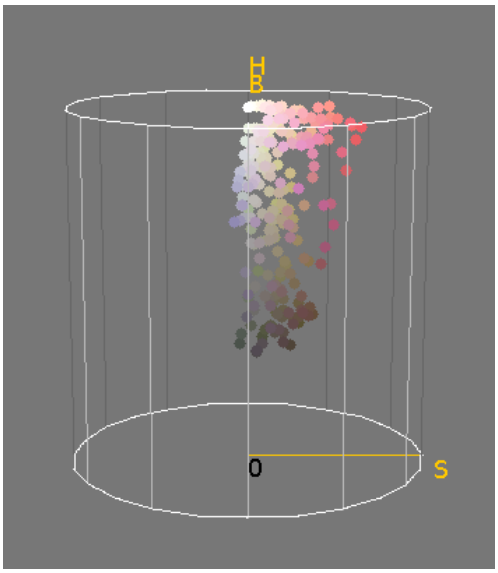


Illustration 27: Cas 1, dalton 42 en HSB.

Dans ce cas, on peut voir que l'image modifié à l'air plus clair que l'image originale. Avec l'image en HSB, on peut voir que la répartition des points et presque identique hormis le fait que dans l'image originale, il y a des points gris foncée en plus, ce qui explique pourquoi elle est plus foncée que l'image modifié.

Conclusion

Durant ce TP, nous avons appris a utiliser plusieurs espaces de couleur pour manipuler une image sur certaines de ses propriétés.

Nous avons ainsi utiliser la représentation HSB pour manipuler la luminance, la saturation et la teinte d'images couleurs grâce à ImageJ. Nous avons également appris à créer des macros pour modifier ces données directement, sans passer par les menus d'ImageJ.

Annexes

Code de la macro « augmentation_luminance » de la partie 1 :

```
macro "augmentation_luminance" {

// recuperation du ID de l'image
image = getImageID();

valeur = getNumber ("quelle augmentation (absolue) de luminance",valeur);

Dialog.create("Debut");
Dialog.addMessage(" Cliquer sur OK pour commencer le traitement ");
Dialog.show();

setBatchMode(true);

// recuperation de la taille W x H de l'image
W = getWidth();
H = getHeight();

run("Duplicate...", "title=luminance modifiee");
image_luminance_aug = getImageID();

for (j=0; j<H; j++) {
   for (i=0; i<W; i++)
      {
      selectImage (image);
      couleur_avant = getPixel(i,j);
      R_avant = (couleur_avant & 0xff0000) >> 16;
      G_avant = (couleur_avant & 0x00ff00) >> 8;
      B_avant = (couleur_avant & 0x0000ff) ;

      //Pour chaque couleur, on ajoute la valeur de luminance entrée par
l'utilisateur.
      //Les ifelse permettent de garder une valeur inférieur à 255.

      if(valeur + R_avant > 255)
            R_apres = 255;
      else
            R_apres = valeur + R_avant;

      if(valeur + G_avant > 255)
            G_apres = 255;
      else
            G_apres = valeur + G_avant;

      if(valeur + B_avant > 255)
            B_apres = 255;
      else
            B_apres = valeur + B_avant;

      couleur_apres = ((R_apres & 0xff ) << 16) + ((G_apres & 0xff) << 8) +
B_apres & 0xff;


      selectImage (image_luminance_aug);
      setPixel(i,j,couleur_apres);
            }
   }
```

```
setBatchMode(false);

Dialog.create("Fin");
Dialog.addMessage(" Cliquer sur OK pour terminer le traitement");
Dialog.show();
}
```

Code de la macro « multiplication saturation » de la partie 2 :

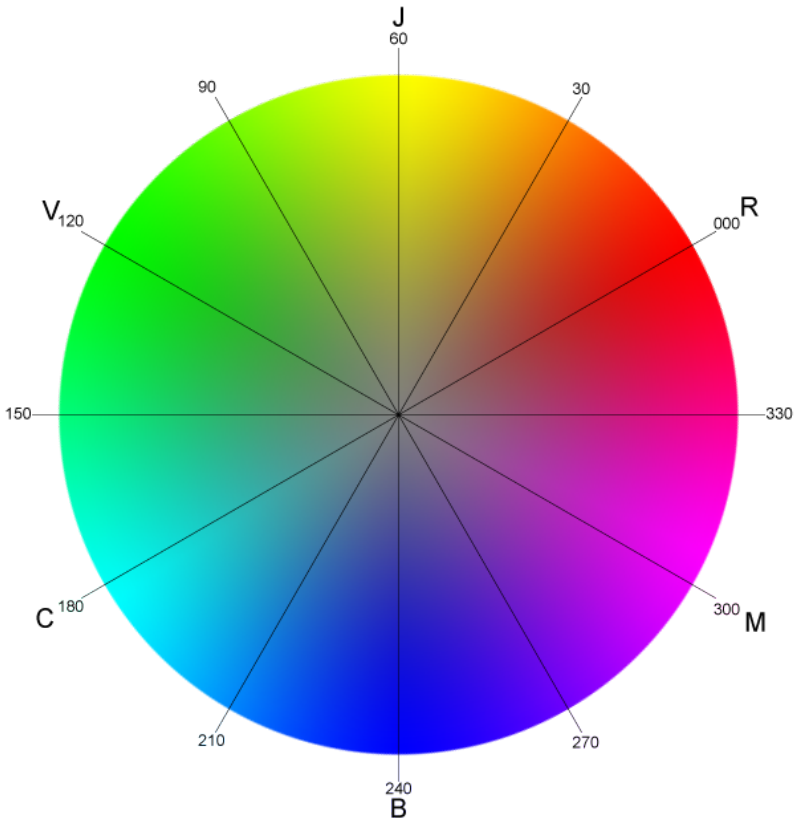
```
macro "multiplication saturation" {

// recuperation du ID de l'image
image = getImageID();

valeur = getNumber ("quelle multiplication de saturation",valeur);

Dialog.create("Debut");
Dialog.addMessage(" Cliquer sur OK pour commencer le traitement ");
Dialog.show();

setBatchMode(true);

titre=getTitle();

run("Color Space Converter","from=RGB to=HSB white=D65");
run("Split Channels");

//Red correspond à la saturation
selectWindow(titre + " (HSB) (green)");
run("Multiply...", "value=" + valeur);

run("Merge Channels...","c1=["+titre+" (HSB) (red)] c2=["+titre+" (HSB)
(green)] c3=["+titre+" (HSB) (blue)] ignore");
run("Color Space Converter","from=HSB to=RGB white=D65");

setBatchMode(false);

Dialog.create("Fin");
Dialog.addMessage(" Cliquer sur OK pour terminer le traitement");
Dialog.show();

}
```

Code de la macro « multiplication saturation » de la partie 3 :

```
macro "modification teinte" {

// recuperation du ID de l'image
image = getImageID();

valeurRed = getNumber ("quelle valeur de rouge ajouter à la
teinte ?",valeurRed);

Dialog.create("Debut");
Dialog.addMessage(" Cliquer sur OK pour commencer le traitement ");
```

```
Dialog.show();

setBatchMode(true);

titre=getTitle();

run("Color Space Converter","from=RGB to=HSB white=D65");
run("Split Channels");

valeurRed = (valeurRed * 256 )/360;
selectWindow(titre + " (HSB) (red)");
run("Add...", "value=" + valeurRed);

run("Merge   Channels...","c1=["+titre+"   (HSB)   (red)]   c2=["+titre+"   (HSB)
(green)] c3=["+titre+" (HSB) (blue)] ignore");
run("Color Space Converter","from=HSB to=RGB white=D65");

setBatchMode(false);

Dialog.create("Fin");
Dialog.addMessage(" Cliquer sur OK pour terminer le traitement");
Dialog.show();
}
```

Cercle des couleurs :


J
60
90
30
V120
000
R
150
330
C
180
300
M
210
270
240
B