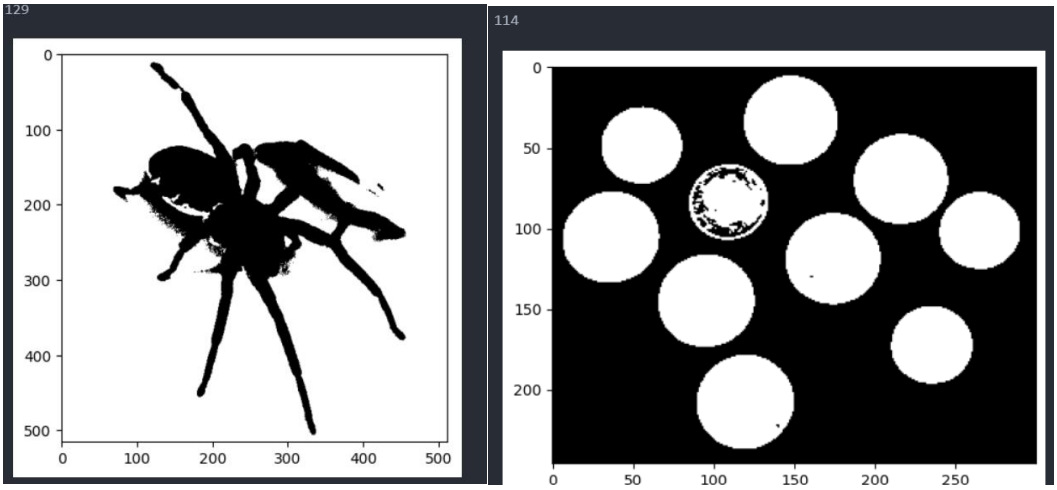
Projet math :

Partie 1 :

Implémentation d'Otsu

Dans le fichier Otsu.ipynb qui me permet de recevoir le seuil qu'on utilise dans la fonction bin2 du tp5 ce qui nous permet d'avoir une image en noir et blanc avec juste le contour des images (par exemple le contour des pièces)



Voilà le résultat de 2 images transformé par Otsu et bin2.

Partie 2 :

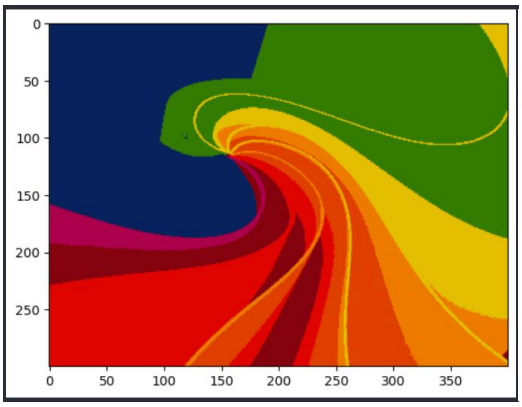
Implémentation du K_means

Dans le fichier K_means.ipynb pour des images en couleurs, noir et blanc et de donner un nombre de clusters.

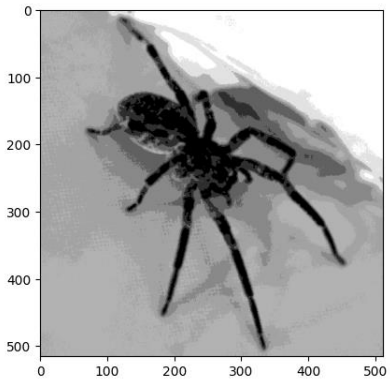
Pour faire cette fonction, il faut prendre le nombre de clusters de manière aléatoire de l'image et ensuite une boucle jusqu'à ce que les clusters ne bougent plus après une itération et ensuite refaire l'image.

Cependant pour lancer une image png il faudrait que j'améliore mon code j'ai déjà fait pour la fonction de détection mais il faudrait que j'améliore en changeant la partie où je recrée l'image mais je n'ai pas su faire.

Voici le résultat avec 2 images une couleur (8 clusters) et un noir et blanc (8 clusters) :



Feuille.jpg



Spider.pgm

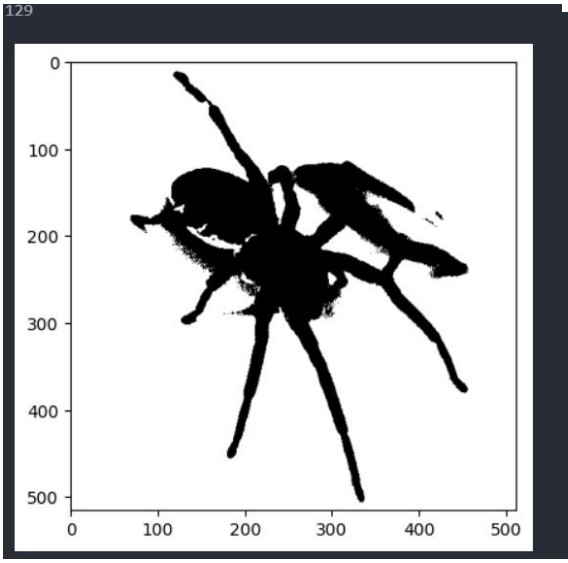
En revanche, je ne sais pas si c'est en rapport avec mon code, pc ou c'est normal (au vu des autres, je dirais que c'est normal) mais il met un temps fou à faire quoi que ce soit.

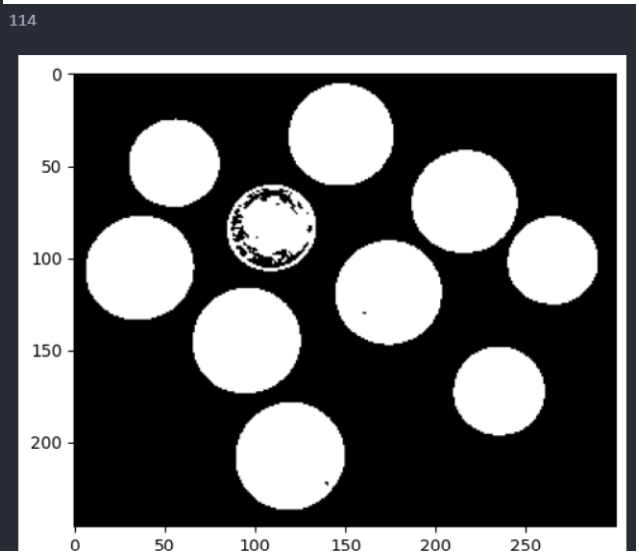
Partie 3 :

K_means est-il meilleur qu'Otsu ?

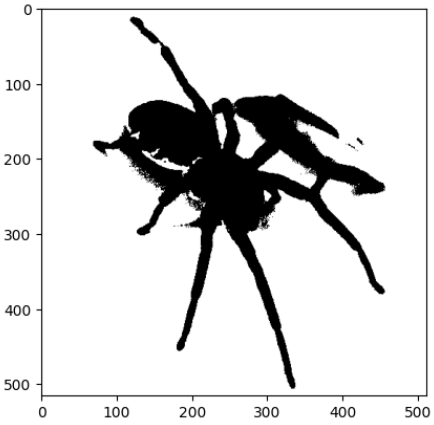
Pour moi pour du noir et blanc Otsu est bien meilleur non pas pour la qualité de l'image, car les 2 me renvoient la même chose, mais surtout en termes de rapidité là ou Otsu le fait instantanément k_means lui met environ entre 1 et 2 min ce qui laisse le temps de boire un bon café contrairement à Otsu.

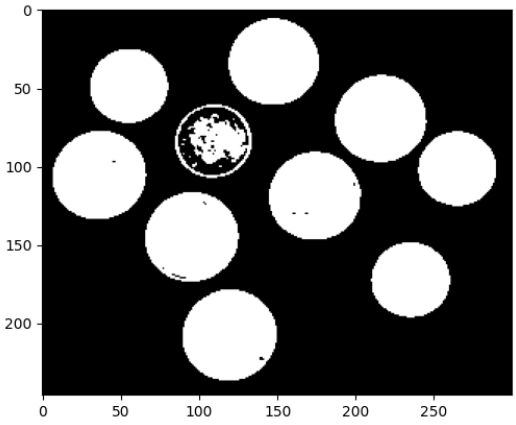
Otsu :





K_means :





En vrai en regardant la pièce k_means on voit que k_means et un chouïa moins bien qu'Otsu (au vu des pixels noir sur certaines pièces.

Partie 4 :

Je n'ai pas eu le temps de faire cette partie