Ambrine Douhane MT-BA1
SCIPER 282708

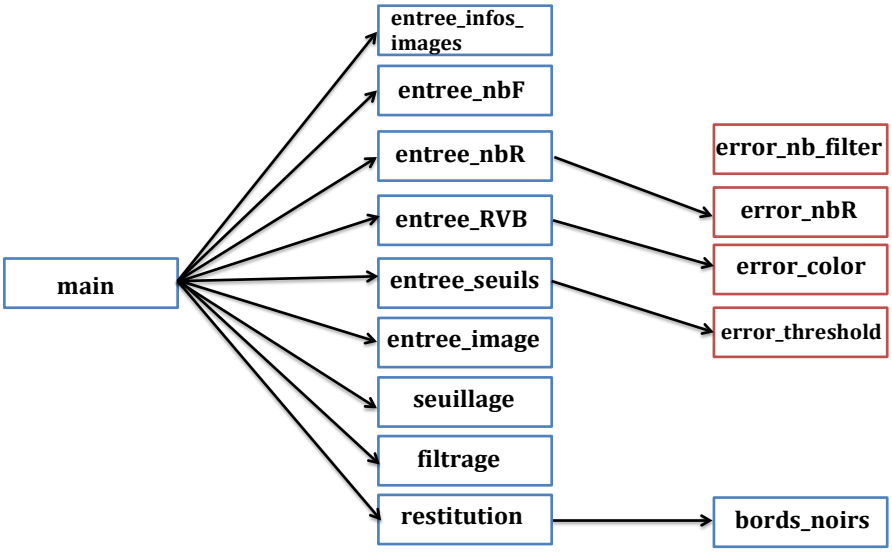
Rapport Color Reduce

1. Phase d'analyse

La fonction main() appelle successivement les fonctions d'entrée et les fonctions de traitement pour effectuer le travail. D'abord, l'utilisateur entre nbR, puis les composantes RVB des couleurs réduites (stockées dans tabl_RVB, dont la permière couleur est le noir(0,0,0)), puis les seuils (stockés dans tabl_seuils). L'utilisateur fournit ensuite au programme le nombre de filtrages nbF, les caractéristiques de l'image et enfin chaque composante R, V, B des couleurs de chaque pixel de l'image à traiter. Toutes ces composantes sont stockées dans un tableau tabl_image dont on se ressert pour afficher l'image en sortie à la fin du programme.
Une fois les informations entrées, la fonction main() travaille avec les fonctions seuillage(), filtrage() et restitution(). J'ai fait le choix d'initialiser les variables dont on se sert dans différentes fonctions dans la fonction main() et de les modifier au fil du déroulement du programme à l'aide du **passage par référence.**

- **La fonction seuillage() permet d'affecter aux intensités (type double) de chaque pixel du tabl_seuillage, les numéros (type int) des couleurs réduites** qui leur correspondent dans chaque pixel du tabl_maj. Pour cela, on se sert de tabl_seuils.
- **La fonction filtrage(), elle, se charge d'affecter à chaque case du tabl_filtrage le numéro d'une couleur réduite en fonction des cases qui l'entourent dans le tabl_maj de mêmes indices.** Comme son nom l'indique, le tabl_maj est mis à jour à la fin de chaque filtrage afin de faire succéder les filtrages à partir de l'image destination. Par ailleurs, afin de déterminer combien de fois, un même numéro i entoure une case du tabl_maj, on se sert d'un tabl_rec qui enregistre la récurrence de i autour de cette case dans tabl_rec[i]. S'il existe un tabl_rec supérieur ou égal à 6, on affecte à la case de tabl_filtrage la valeur de l'indice de la case qui le contient. Autrement, la case du tabl_filtrage vaut 0. Aussi, la fonction filtrage() appelle la **fonction bords_noirs() à la fin de cette opération. Elle a pour but d'affecter des 0 cases extérieures de tabl-maj.** Ici, on utilise le passage par référence du tabl_maj.
- Une fois le filtrage opéré, la fonction **restitution() affecte dans tabl_im les composantes R, V et B des couleurs réduites aux pixels** alors sous la forme du numéro de leur couleur dans tabl_RVB puis **affiche** quelques caractéristiques importantes de l'image voulue ainsi que l'image elle-même.

Le graphe ci-dessous montre les liens entre la fonction main() et les autres fontions.



2. Pseudo-code de l'étape de filtrage

Entrées :

nbF, tabl_maj

Traitements :

```
Affecter des 0 à toutes les cases de tabl_rec
nouv_coul ← 0
pour f allant de 1 à nbF
{
        pour chaque case tabl_maj[i][j]
        {
                pour chaque case tabl_maj[i'][j'] autour de tabl_maj[i][j]       //càd i' != i ou j' != j
                        tabl_rec[tabl_maj[i'][j']] ← tabl_rec[tabl_maj[i'][j']]+1

                pour coul allant de 0 à nbR-1
                        si tabl_rec[coul] >= 6
                                nouv_coul ← coul

                tabl_filtrage[i][j] ← nouv_coul
                nouv_coul ← 0
                Affecter des 0 à toutes les cases de tabl_rec
        }

        tabl_maj ← tabl_filtrage
        appel de la fonction bords_noirs
}
```

Sortie :

3. Complexité de l'algorithme de filtrage

En supposant que nbR et nbF sont très petits devant nbC et nbL, **la complexité de ce code est O(nbC x nbL).** En effet, pour chaque filtrage, les opérations effectuées autour de nbC x nbL cases ont un coup constant. Ces opérations sont :

- 6 affectations d'entiers dans tabl_rec : O(6)
- nbR comparaisons d'entiers contenus dans tabl_rec avec 6 : O(nbR)
- 1 affectation d'entier dans tabl_filtrage : O(1)

On a ansi que pour chaque case, la complexité des traitements vaut O(6.nbR.1) qui est une complexité constante.