H.E.P.L. : Catégorie Technique

Bachelier en Informatique et Systèmes : finalité Informatique Industrielle

LABORATOIRE D'INFORMATIQUE INDUSTRIELLE

Traitement d'image

3ème année

Etape 1

2012 - 2013

Van Gysegem Thomas

Groupe : 2322

Table des matières

Explication du fonctionnement :

MainWindows

Cette classe instancie la fenêtre de base de l'application, elle définit le comportement des différents boutons, menus et autres boîtes de dialogues . Elle permet d'afficher les images et de lancer les traitements désiré

Picture

Cette classe permet la manipulation d'une image donnée. Elle implémente les traitements demandé dans le cahier des charges (ROI, Taille, Palette et d'autres à venir)

Dialog*

Les classes commencant par Dialog implémentent les boîtes de dialogue et la facon dont elles interagissent avec la fenêtre principale. Pour l'instant, elles sont au nombre de deux : DialogModifTaille et DialogModifPalette.

DialogModifTaille

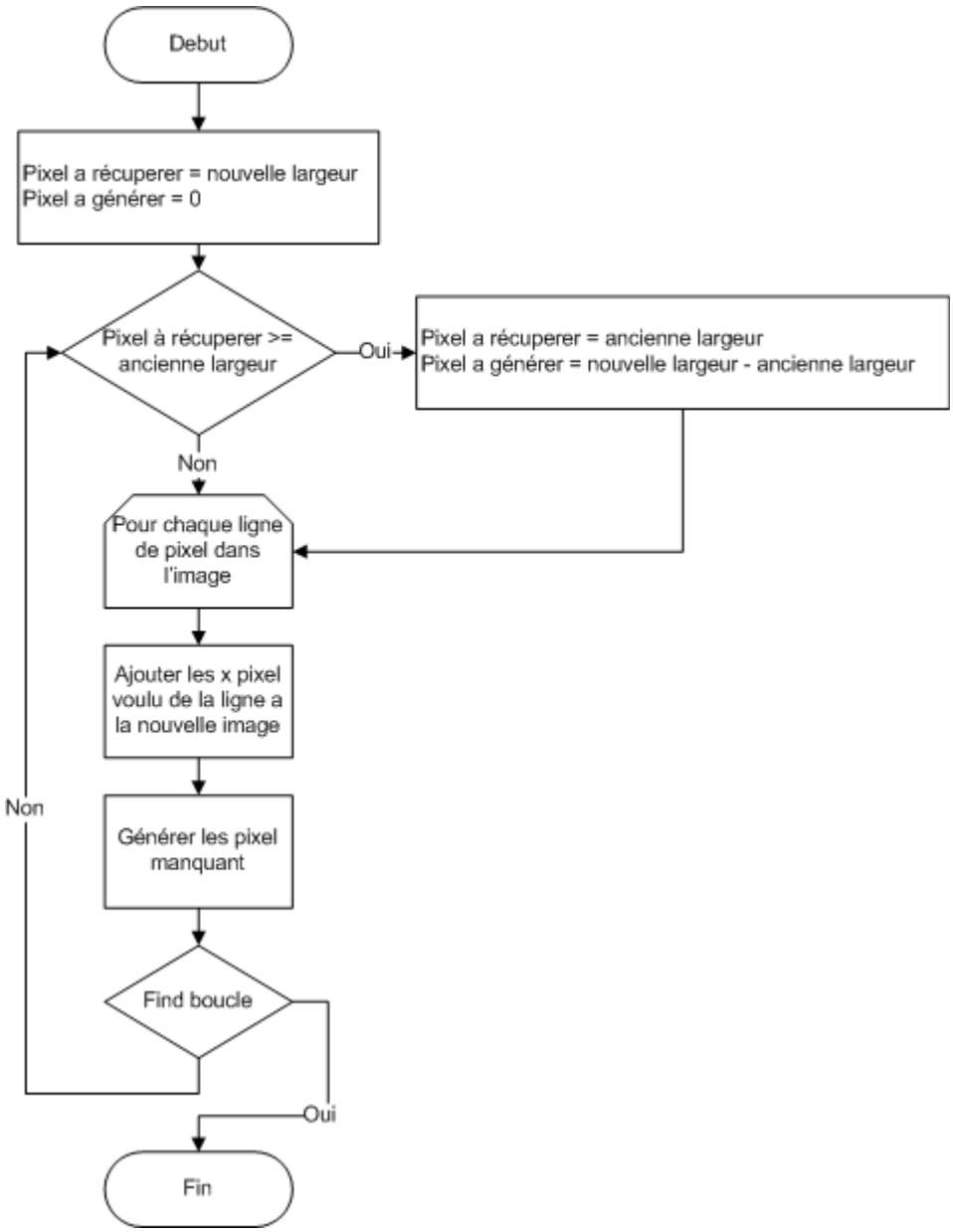
Se charge de fournir une interface a l'utilisateur permettant de modifier la taille de l'image

DialogModifPalette

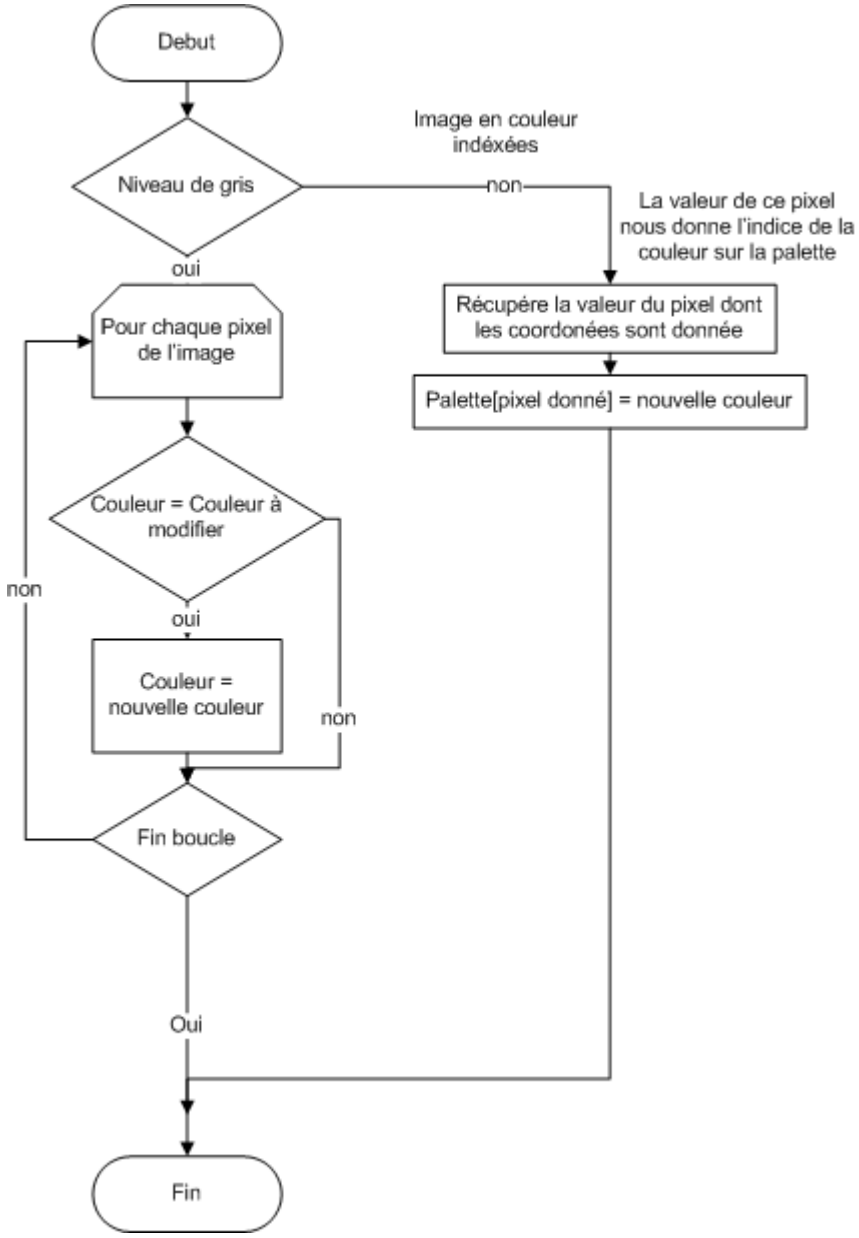
Se charge de fournir une interface a l'utilisateur permettant de modifier une couleur de la palette de l'image

Schéma des processus :

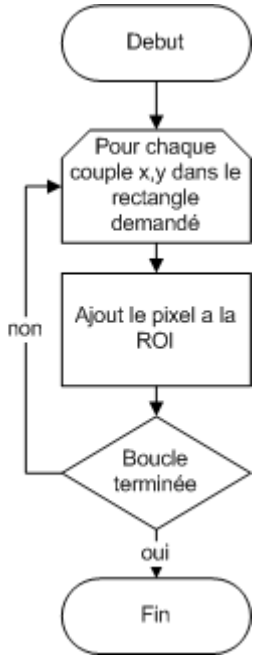
Modification de la taille



Modification de la palette



Récupérer la ROI



Justification des choix :

J'ai choisis le Python car c'est un langage qui, selon moi, permet de mettre en œuvre des algorithmes complexe ou non en très peu de temps. De plus, c'est un langage interprété, les test lors du développement sont donc beaucoup plus rapide car ils ne nécessitent pas de compilation longue et fastidieuse.

La GUI utilisée est créée via la librairie PyQt. Cette librairie possède un éditeur de GUI très performant et facile d'utilisation. Pour charger les images j'utilise la librairie PIL, relativement bas niveau comparée à d'autres comme OpenCV.

Tout ces choix me permettent donc de développer rapidement et efficacement l'application demandée. Le seul bémol de Python étant qu'il n'est pas le plus rapide des langages et que donc, certains algorithmes pourraient être plus chronophage en Python que dans d'autres langage.

Commentaires :

Lors de cette première étape j'ai rencontré beaucoup de difficultés lorsque j'ai voulu afficher mon image avec Qt. La plupart des documents en ligne indiquaient d'utiliser un Qlabel pour afficher une image mais ce dernier ne permettait pas de sélectionner une zone directement sur l'image comme demandé dans le cahier des charges. J'ai donc dût me familiariser avec le concept de QgraphicView/Scene qui permet de gérer des objets en 2D dans une scène et pour lequel l'objet QgraphicView fait office de caméra.

J'ai également rencontré des problèmes lors de la modification de la palette de l'image. En effet, la documentation officielle de la librairie PIL ne spécifiait pas l'existence d'une fonction permettant de récupérer la palette chargée avec l'image (pour les images indexées).

On peut aussi mentionner l'optimisation de la modification de la taille de l'image, la première façon de le faire à laquelle j'ai pensé prenait énormément de temps (un simple parcourt de « nouvelle largeur x nouvelle hauteur » en allant rechercher les pixel voulu dans l'image source) et j'ai donc rapidement dût l'optimiser en copiant directement des lignes entières de pixel de l'image originale et en générant les pixel manquant, je suis donc passé de deux boucles imbriquée à une seule boucle sur la hauteur de l'image. Cette dernière méthode reste suffisamment rapide pour des images jusqu’à environ 2000x2000 pixel de dimension.