Rapport de projet

Auteurs

- Huot de Saint Albin, Raphael, raphael.huot-de-saint-albin@universite-paris-saclay.fr, LDDMP2

Résumé du travail effectué

- Partie 2: Traitée La partie 2 a été traitée et fonctionne entièrement.
- Partie 3: Traitée La partie 3 a été traitée et fonctionne entièrement.
- Partie 4: Traitée La partie 4 a été traitée et fonctionne entièrement. Le lissage est un flou gaussien (avec un noyau 3*3). La fonction doubleSeuillage s'exécute tant que des améliorations sont apportées. Toutes les questions ont été traitées sauf la 1ere question de la 4.8, les optimisations n'étant finalement pas rentables.
- Partie 5: Traitée La partie 5 a été traitée et fonctionne entièrement.
- Partie 6: Traitée La partie 6 a été traitée et fonctionne entièrement.
- Partie 7: Traitée La partie 7 a entièrement été traitée. Pour mieux visualiser l'image, les bords bleus sont affichés mais la couleur des pixels n'est pas modifiée (code en commentaire).

L'intégralité des améliorations proposées a été réalisée. Toutes les fonctions de lecture et d'écriture disposent d'une gestion efficace et complète des commentaires. Toutes les fonctions sont commentées et documentées. De nombreux tests sont réalisés (en partie grâce a la macro TEST) afin de vérifier que les entrées ne soient pas vides. En cas d'erreur critique, des exceptions détaillées sont lancées. En cas d'erreurs non fatales, des warnings sont affichés sur la sortie d'erreur du terminal (std::cerr). Des améliorations non proposées ont été implémentées, comme par exemple la fonction filtreConvolution, prenant en entrée un noyau et une image et appliquant ce filtre à l'image, ou encore la fonction fusionGrisCouleur, transformant 3 images en nuances de gris en une image en couleur. Les fonctions prenant un tableau à double entrée utilisent la macro TESTIMAGE pour vérifier que les entrées ne sont pas vides.

Démonstration

Je montrerai la compilation et le lancement de l'ensemble des tests. Pendant l'execution (2mn), je detaillerai plusieurs aspects de mon code:

- Quelques détails de la fonction de lecture (gestion des commentaires).
- Une présentation détaillée, code a l'appui, de la fonctionalité des superPixels.
- Une explication concise de la convolution de matrices (cas général).
- Mention des fonctions de fusion et de découplage des images en couleur.
- Une présentation rapide des résultats de doubleSeuillage et de superPixels.

Lors des questions, les points les plus interessants à étudier sont les détails de ce

qui vient d'être énoncé. En particulier, il peut être interessant de détailler le code des fonctions de lecture et leur gestion des commentaires et des erreurs. Un autre point interessant est l'étude de la variation des résultats lors de la modification des différents paramètres des fonctions doubleSeuillage et superPixel. Enfin, la demonstration de tests de convolution peut également être pertinente.

Organisation du travail

Pour travailler, j'ai beaucoup utilisé git avec GitLab pour pouvoir gérer les sources de mon projet, pouvoir retrouver mes anciens codes et transferer le code entre mes différents équipements. J'ai passé une quarantaine d'heure sur le projet, afin d'obtenir le rendu le plus propre possible. Mon objectif était d'avoir un code type API, utilisable facilement mais sécurisé, et effectuant des vérifications au niveau des entrées.

Prise de recul

J'ai appris énormément sur le traitement d'image de manière générale. Si je devais refaire ce projet, j'aurai créé un projet plus générique, avec une seule fonction de lecture, des structures de données, des classes. J'aurai beaucoup plus orienté objet ce programme et je l'aurai plus conçu comme une API plutôt que comme un logiciel en soi.