Rapport

Projet Programmation

IMAGIMP

IMAC 1

Antoine DEMIERE & Lisa COUAVOUX

Sommaire

1. Présentation du projet
2. Fiche travaux
3. Organisation du projet
4. Architecture du programme
5. Analyse détaillée
6. Difficultés rencontrées
7. Conclusion

1. Présentation du projet

Ce projet a pour but d’aborder quelques notions clés utilisées par les logiciels classiques de traitement d'images comme Photoshop ou Gimp, indispensables de nos jours pour l'édition multimédia.

A travers des éléments clefs de logiciel semblable tels que les claques, les LUT (Look-up Tables) ou encore les histogrammes, nous avons implémenté en langage C une interface afin de procéder au traitement d’image désiré.

Etant donné le cahier des charges, le peu de temps et la multitude de projets parrallèle à celui-ci, nous avons élaboré une collaboration avec le binôme de Matthieu Bessol et Baptiste Olivier. Grâce à cette collaboration, nous avons pu mettre en place une IHM fonctionnelle en utilisant les librairies OPENGL étudié en cours de synthèse d’images.

1. Fiche Travaux

Éléments demandés et codés qui fonctionnent : (voir l’analyse détaillée)

* Des calques
* Des LUT
* Des effets (sépia, noir et blanc)
* Un histogramme

Éléments demandés mais pas codés :

* Le calcul de l’histogramme est fait directement à partir de l’image et non selon les LUT.

Éléments non demandés et codés qui fonctionnent :

* Une IHM qui rend le programme fonctionnel

Éléments non demandés mais pas codés ou qui ne fonctionnent pas

III. Difficultés rencontrées

Les principales difficultés rencontrées dans le développement de l’application ont été :

* L’implémentation des fusions de calques et de LUT
* L’IHM a aussi été une source de difficultés
* Nous avons des difficultés concernant la suppression des calques qui entraine la suppression du bouton associé mais des problèmes de liaison peuvent avoir lieu entre les boutons et les calques. Le bouton sélectionné ne correspond plus forcément au calque associé.
* Le chargement d’image n’est pas totalement fonctionnel. Nous avons un problème concernant les droits d’accès au fichier.
* L’organisation du projet en parallèle avec les autres projets et partiels !

IV. Organisation du projet

Le projet s’organise selon différent fichier :

Bin : contient l’exécutable (imagimp.exe)

Include : cofhnntient les fichiers .h

Src : contient les fichiers .c

Obj : contient tous les .o

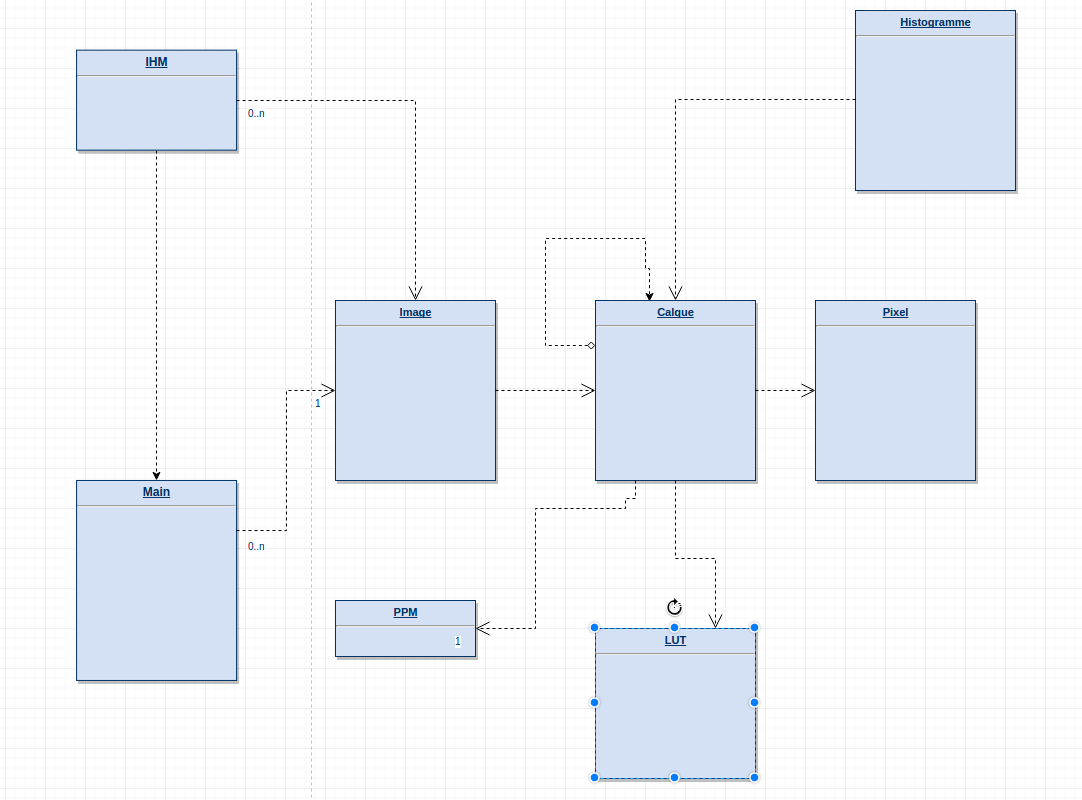
Images : contient toutes les images .ppm

Doc

Un Makefile est opérationnel sous les trois OS (Linux, Windows, Mac).

Git a été utilisé pour organiser le projet.

V. Architecture du programme

GRAPHE UML  
  
  
  


VI. Analyse détaillée

L’image

typedef struct Image{

Calque\* listCalques;

Calque\* calque\_resultat;

} Image;

Une image PPM est chargée à l’ouverture du programme en récupérant la chaine de caractère des pixels (RVB) qui se trouve

La sauvegarde de l'image finale se fait par défaut en save.ppm qui se trouve dans le répertoire image.

Autres fonctions présentes :

* Fusion de calques et LUT
* Application du LUT sur un calque
* Application des efffets sur le calque
* Dessin de l’histogramme
* Affichage du calque

Toutes ces fonctions prennent un pointeur sur la structure Image (et un autre paramètre pour certaines d’entre elles) et renvoient un void. L’image passée en paramètre est traitée directement dans les fonctions.

typedef struct Calque{

Fusion fusion;

Pixel\*\* pixels;

LUT\* listLuts;

float alpha;

struct Calque\* next;

struct Calque\* prev;

int height, width;

Histogramme\* histogramme;

unsigned int id;

int isSelected;

Effet effet;

} Calque;

Le calque

* Il est possible **d’ajouter un calque** en appuyant sur le bouton « Nouveau calque » de l’IHM.
* **La** **navigation** dans les calques se fait au clic en dessous de l’image chargée. L’utilisateur peut ainsi savoir où il se situe dans les calques, puisque ces derniers sont numérotés par ordre croissant d’ajout.
* Le **paramètre d’opacité** d’un calque se fait sur le calque final (image fusionnée).
* La suppression d’un calque se fait.

Les calques sont répertoriés par des ID (int).

Une liste doublement chainée est implémentée.

Chaque calque a la même dimension.

Les calques se combinent pour crééer l’image finale.

L’histogramme

L’histogramme illustre l’image selon des valeurs allant de 0 à 255. Il existe trois histogrammes regroupés : un pour le rouge, un pour le vert, un pour le bleu et une moyenne de trois pour les niveaux de gris.

typedef struct Calque Calque;

typedef struct histogramme{

int valeursRed[256];

int valeursGreen[256];

int valeursBlue[256];

int valeursLuminosite[256];

int valeurMaxBlue;

int valeurMaxRed;

int valeurMaxGreen;

int valeurMaxLum;

int valeurMax;

}Histogramme;

Les LUT (Look-up Table)

typedef struct LUT{

int lut[256];

struct LUT\* prev;

struct LUT\* next;

LutOption type;

}LUT;

* L’ajout d’un LUT
* Suppression des LUT

2. Appliquer une LUT a une image (LUT\_2), 3. Supprimer la derniere LUT (LUT\_3). Les LUT que l'application devra pouvoir ajouter sont listées ci-dessous. Chacune de ces modifications possède un code indiqué entre parentheses. augmentation de luminosité (ADDLUM), dépend d'un paramètre diminution de luminosité (DIMLUM), dépend d'un paramètre augmentation du contraste (ADDCON), dépend d'un paramètre diminution du contraste (DIMLUM), dépend d'un parametre inversion de la couleur (INVERT), effet sepia (SEPIA), peut dépendre d'un ou plusieurs paramètres. L'ajout d'une LUT se fera toujours en fin de liste de LUT pour un calque donné. L'ajout se fait sur le calque courant. L'application d'une LUT à une image est detaillee dans le paragraphe 1.5. La suppression d'une LUT est effectuée a la fin de la liste. Elle se fait sur le calque courant.

Les effets

Pour supprimer un effet, il suffit d’appuyer sur un autre effet. Ainsi, on ne peut pas superposer plusieurs effet.

Pour annuler

L’IHM

Elle est composée de structure de boutons, slider, bouton de calque et une structure IHM regroupant toutes les fonctionnalités de l’IHM.

Le dessin de l’interface dans son ensemble est réalisée avec OPENGL, ce qui mène à une interface originale. (utilisation de glVertex()).

L’application permet à l'utilisateur d'en sortir, en appuyant sur la croix. La sortie se fait proprement (désallocation de la mémoire).

• D'autre part, pour effectuer des modications pertinentes sur l'image, on exploite fréquemment l'histogramme de l'image finale. Cet histogramme apporte des informations importantes sur l'image comme son niveau de contraste et son exposition.

• Des Look Up Table (LUT) permettent ensuite de modifier l'histogramme de cette image, dans le but de réaliser des effets.

• Enfin un bon logiciel d'édition d'image offre toujours la possibilité de défaire et de refaire les actions de l'utilisateur. Le but de ce projet est d'implémenter en C ces quatre éléments fondamentaux des logiciels de traitement d'images.

VII. Conclusion

Ce projet nous a finalement permis de nous familiariser avec d’une part les quelques outils indispensables du traitement d'images, utilisés dans les logiciels concernés et d'autre part, l'implémentation en C des images.

Par ailleurs, ce projet nous a permis de développer des aptitudes concernant le travail en groupe. Nous avons utilisé git pour synchroniser les modifications de chacun.

Comme ouverture, nous pouvons proposer une gestion de l’historique, l’ajout d’effets tels que Nashville ou autres filtres instagram. De plus, pour optimiser la précision du traitement d’image, un zoom aurait pu être introduit.