Projet C – IMAC 1
Logiciel Imagimp

Katia Moreira – Audrey Guénée

Janvier 2012

Introduction

Imagimp est un petit logiciel de traitement d'images PPM codé en C et s'appuyant sur une interface graphique simple (Glimagimp). Ce projet avait pour but d'améliorer nos compétences en programmation grâce au langage C et de découvrir les notions clés utilisées par les logiciels de traitement d'images.
Ses principales fonctionnalités sont l'ajout de calques, la modification de l'opacité et du type de mélange de ces calques et l'application de LUT.

Ce court rapport présente, de manière non exhaustive, nos réflexions sur les structures de données et les fonctions implémentées pour réaliser le projet.

Bien que le projet ne soit pas totalement fini, nous tenons à signaler qu'aucun ordinateur n'a été endommagé (ou maltraité)*, ce qui est déjà pas mal...

Bonne lecture !

*(Mais notre moral, OUI.)

I. Structures de données

1- Image

Imagimp ne peut ouvrir que des images PPM binaire (de type P6).
Contenu d'un fichier PPM binaire
P6
Commentaires
largeur
hauteur
valeur maximale pour un niveau de couleur (généralement 255)
Pixels de l'image

La structure *Image* reprend donc toutes les données contenues de le fichier.
Structure Image

```
typedef struct Image {
   unsigned int type;
   unsigned int height;
   unsigned int width;
   unsigned int max;
   unsigned char* pixel;
} Image;
```

A chaque image PPM chargée en mémoire (donc, ouverte dans le programme), est associée un calque d'opacité 1.0 et de mélange multiplicatif (valeurs par défaut).
Aussitôt l'image PPM initiale ouverte, un calque reprenant les pixels de l'image est créé : ce sont les pixels de ce calque qui apparaissent à l'écran (dans l'IHM) et non ceux de la structure *Image*.
La structure *Image* n'est utilisée que pour l'ouverture de l'image initiale et la création de l'image finale.

2 - Calques

Ce sont sur les calques que sont appliquées les modifications comme le changement d'opacité, le type de mélange et l'ajout de LUT.
Il y a deux types de calque : les calques vides et les calques images.

Structure Layer

```
typedef struct Layer {
  unsigned int id;
   struct Image* source;
   struct Layer* prev;
   struct Layer* next;
   float opacity;
   unsigned int mix;
   struct Lut* appliedLut;
   unsigned char* pixel;
} Layer;
```

Les calques dépendent d'une image source :
- une image complètement blanche pour un calque vide
- une image PPM pour un calque image
La structure calque possède un lien vers cette image source (struct Image* source) qui permet d'avoir accès aux informations de l'image originale via le calque.
Comme la structure *Image*, le calque a son propre tableau de pixels (unsigned char* pixel) qui contient soit :
- les valeurs originales des pixels de l'image source, c'est le cas lorsque aucune modification n'a été appliquée (calque->pixel = calque->source->pixel)
- les valeurs re-calculées des pixels de l'image source après modification de l'opacité et/ou du type de mélange et/ou l'application d'une LUT.

Finalement, l'image ouverte n'est jamais modifiée : le tableau de pixels de la structure *Image* créé à son chargement reste inchangé. C'est le calque auquel est associé l'image qui est modifié. Autrement dit, c'est le tableau calque->pixel qui est re-calculé et non calque->source->pixel.
L'intérêt d'une telle structure est de pouvoir toujours avoir accès à l'aperçu de l'image source telle qu'elle a été ouverte dans le programme (mode de vue *Image source*).

3- LUT

Les LUT permettent d'appliquer des effets aux calques (luminosité, niveaux de gris, sépia, contraste…).
Une LUT associe à chaque pixel, une valeur de sortie qui modifie l'apparence de l'image.
Chaque niveau de couleur (Rouge, Vert, Bleu) peut prendre 256 valeurs différentes.
Par exemple, la case 0 du tableau tabR[256] contient la valeur de niveau de rouge que prendra tous les pixels de l'image d'entrée ayant un niveau de rouge à 0.

Structure LUT

```
typedef struct Lut {
  int type;
  unsigned char tabR[256];
  unsigned char tabV[256];
  unsigned char tabB[256];
  struct Lut* next;
} Lut;
```

Contrairement au changement d'opacité, les pixels du calque ne sont pas re-calculés à partir des pixels de l'image source à l'application d'une LUT : c'est-à-dire que les LUT précédemment appliquées au calque sont prises en compte si une nouvelle LUT est appliquée (ex: si on diminue la luminosité d'une image, puis qu'on lui applique l'effet sépia, l'image finale visible à l'écran correspondra à la combinaison des deux LUT).

La luminosité

Concernant l'augmentation et la diminution de la luminosité, nous avons implémenté deux algorithmes simples. On demande simplement une valeur comprise entre 0 et 255 à l'utilisateur ; dans le cas de l'augmentation de la luminosité, on ajoute cette valeur aux cases de nos tableaux de LUT, dans le cas de la diminution de la luminosité on la soustrait

Le contraste

Pour modifier le contraste de l'image on a joué sur le fait que les couleurs claires ou foncées étaient plus ou moins "proches". Dans les deux cas on a demandé à l'utilisateur de rentrer une valeur entre 0 et 255.
Pour l'ajout de contraste, on a soustrait la valeur entrée par l'utilisateur à la valeur du tableau de LUT si celle-ci était inférieure à 128. Sinon on l'a ajoutée. Ainsi les valeurs inférieures à 128 étaient d'autant plus diminuées, les valeurs supérieures augmentées afin que l'écart entre les couleurs claires et foncées soit plus important.
Pour la diminution de contraste on a fait en sorte que les valeurs du tableau tendent vers la valeur moyenne 128. On a donc augmenté les valeurs inférieures à 128 en faisant attention qu'elles ne dépassent pas ce seuil et diminué les valeurs supérieures à 128 sans qu'elles ne deviennent inférieures. Ainsi l'écart entre les couleurs claires et foncées diminue et l'image tend globalement vers un teint "grisâtre".

L'effet sepia

Pour l'effet sépia nous avons tout d'abord réalisé l'effet niveau de gris. Cet effet n'utilise pas de LUT puisqu'il s'applique directement aux pixels de l'image. Pour obtenir un niveau de gris, on prends les valeurs R, V, B de chaque pixel et on les divise par 3 ; on applique la valeur ainsi obtenue aux 3 composantes. Elles ont donc alors la même valeur, notre pixel est donc gris.
L'effet sépia va consister à passer notre image en niveau de gris puis à modifier son "teint" de telle sorte qu'elle prenne un teint sépia. Ainsi sur la base de nos pixels en niveau de gris on va ajouter la valeur de 80 aux composantes rouges, la valeur de 30 aux composantes vertes et retrancher 40 pour le bleu.

Inversion des couleurs

L'inversion des couleurs se fait grâce à un calcul simple. Il suffit de retrancher à 255 la valeur contenue dans le tableau de LUT. Ainsi la case 0 par exemple (noir) deviendra 255-0 = 255 (blanc), à l'inverse la case 255 (blanc) deviendra 255-255 = 0 (noir) ; on a bien une inversion de couleurs.

4 - Historique

Structures maillonHistorique et pileHistorique

```
typedef struct maillonHistorique {
  Calque* calque;
  maillonHistorique* next;
} maillonHistorique;

typedef struct pileHistorique {
  maillonHistorique* first;
  int taille;
} pileHistorique;
```

II. Fonctionnalités

1 - Côté image

✓ Charger une image PPM
✓ Sauvegarder l'image finale

L'image finale correspond au dernier calque de la pile, celui qui est au-dessus de tous les autres : ses pixels sont la synthèse de toutes les modifications apportées aux calques précédents et donc le résultat final de l'image.
Ce sont donc les pixels de ce dernier calque qui sont enregistrés dans le tableau de pixels de l'image finale.

2 - Côté calque

✓ Ajouter un calque vierge
✓ Naviguer dans les calques

L'affichage des calques dans le menu latéral de l'IHM est peu fiable : affichage dans le désordre et id des calques erronés.

✓ Modifier le paramètre d'opacité d'un calque

Si un calque C est modifié, le calque C+1 au-dessus de ce dernier, ne sera plus à jour : son aperçu dans l'IHM correspondra au calcul précédemment effectué, avant la modification du calque C (avec les valeurs initiales des pixels du calque C).

✓ Modifier la fonction mélange du calque
✓ Supprimer le calque courant

3 - Côté LUT

✘ Ajouter une LUT
✓ Appliquer une LUT à une image
  ✓ augmentation/diminution de luminosité
  ✓ augmentation/diminution de contraste

- ✓ inversion de la couleur
- ✓ effet sépia
- ✓ effet noir et blanc
- ✓ effet inversion des couleurs

Si on modifie l'opacité du calque ou son type de mélange après avoir appliqué une LUT, cette dernière n'est plus appliquée à l'image car les pixels sont recalculés à partir de l'image source (donc sans la LUT).

✘ Supprimer la dernière LUT

4 - Côté IHM

✓ Changer de mode de vue

Il y deux modes de vue :
image finale (par défaut)
image source
Le mode de vue image finale permet de voir toutes les modifications apportées aux calques (opacité, mélange, lut).
Le mode de vue image source permet de voir les images sources originales des calques avant toutes modifications (telles que les images ont été ouvertes dans le programme).

✓ Charger une première image en ligne de commande
✘ Appliquer des effets sur ce premier calque en ligne de commande
✓ Sortir de l'application

5 - Côté Historique

✘ Afficher l'historique des modifications
✘ Annuler la dernière opération si possible

III. Notice d'utilisation

1 - Pour exécuter le programme

Réinstaller la bibliothèque glimagimp (à chaque fois)
Dans le répertoire ProjetC/lib/glimagimp, compiler avec make et installer avec . ./install.sh

Exécution de glimagimp
Dans le répertoire ProjetC, compiler avec make
Dans le répertoire ProjetC/bin, exécuter avec ./imagimp

2- Touches spéciales du programme

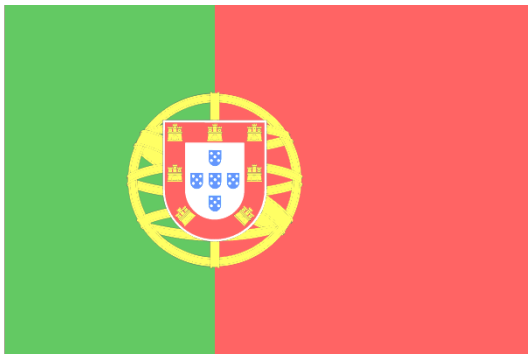
Un menu principal et des sous-menus associent chaque opération possible dans le programme à une commande du clavier :

```
---------- Menu principal ----------
  [i] Afficher les informations de l'image
  [1] Accéder au menu calque
  [2] Accéder au menu LUT
  [3] Changer de mode de vue
  [4] Sauvegarder l'image finale
------------------------------------------------------
```

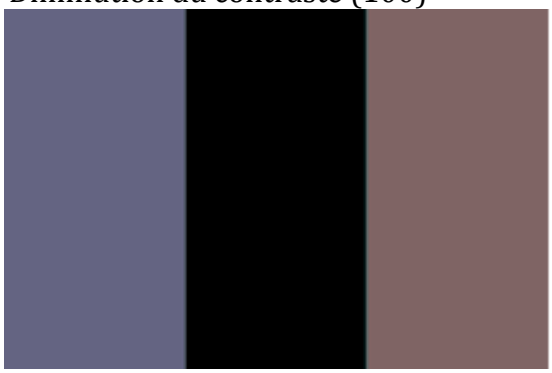
Pour les autres touches :
(echap) Fin du programme
(f. haut) Aller sur le calque au-dessus du calque courant
(f. bas) Aller sur le calque en-dessous du calque courant

IV. Résultats

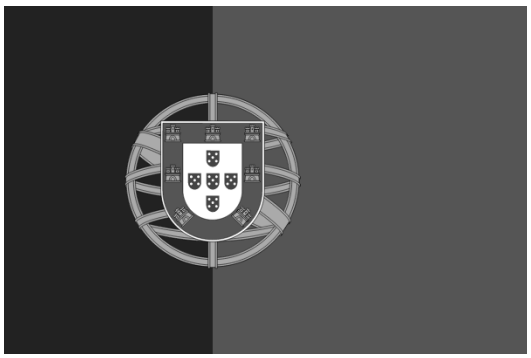
Augmentation de la luminosité (100)



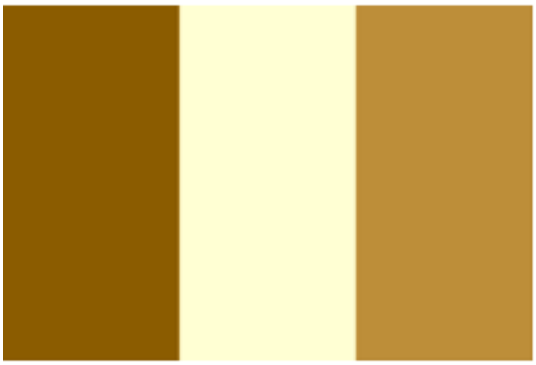
Diminution du contraste (100)



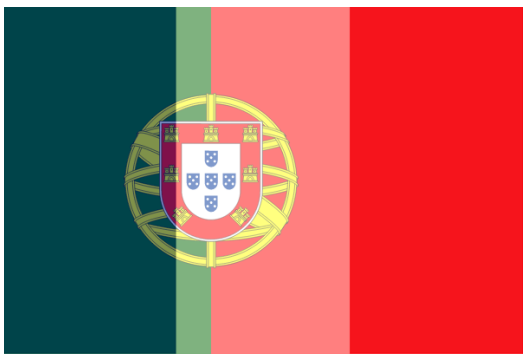
Noir et blanc



Sépia



Diminution opacité (0.5)



Diminution opacité 0.5 et mélange additif