DECAMPS Marceau
LAGRANGE Nathan
Binôme 4

CR Projet CPS

Traitement d'image au format ppm

Ce projet nous a été donné afin de clôturer le semestre en « C pour la Programmation Système ». Le but est de traiter des images au format ppm (portable pixmap) pour les rendre au format pgm (portable graymap) ou au format pbm (portable bitmap). La différence sur ces format est simplement le stockage des pixels. En effet, le format ppm prend 2 octets pour chaque couleur rouge, vert et bleu par pixel. Le format pgm quant à lui ne prend que 2 octets par pixel, ce qui donne une interprétation de l'image en nuances de gris. Enfin, le format pbm prend qu'un seul bit par pixel, ce qui donne une image en noir & blanc, sans nuances.

Structures et fonctions

Nous n'avons fait qu'une seule structure pour gérer l'ensemble des formats (visible dans le fichier */Application/the_headers/struct.h*). Cette structure contient toutes les données dont on a besoin pour traiter l'image, soit le type du format, la largeur (en pixel), la hauteur (en pixel), la valeur max d'un pixel et enfin un tableau contenant tous les pixels. Ce dernier est un tableau d'**unsigned int sur 64 bits**. Prendre une variable avec 64 bits nous permet de gérer chaque pixel, peut importe le format, sur qu'une seule variable. Par exemple, pour une image au format ppm, nous allons coder la couleur bleu sur les bits 0 à 15, la couleur verte sur les bits de 16 à 31, et enfin la couleur rouge sur les bits 32 à 47.

Au niveau des fonctions, nous avons plusieurs fichiers pour répondre à tous nos besoin de manière structuré. Tout d'abord nous avons le fichier *util.c* qui regroupe les fonctions utilitaires pour le fonctionnement du main. On commence donc avec la fonction `char *lignecommande(int argc, char *argv[], int* b, int* g, FILE **fichier)` qui nous permet de décoder la ligne de commande et de trouver les bons arguments. Dans cette fonction nous regardons chaque argument puis, si c'est un paramètre (-b ou -g) nous incrémentons les variables *b* ou *g*. et si ce n'est pas un paramètre, on considère que c'est un nom de fichier alors on essaye de l'ouvrir. Dans le cas ou nous arrivons pas a ouvrir le fichier, ou si il n'y a pas de nom de fichier, on envoi redirige le fichier sur la variable *stdin* (entrée standard du prompt). Pour finir, cette fonction retourne un char* qui contient le nom du fichier passé en paramètre, ou un nom par défaut si nécessaire.

Nous avons ensuit une fonction `struct structimg lectureFichier(FILE* fichier)` qui va retourner un structure de type *structimg* (décrite plus haut) en remplissant tous les paramètres en fonction du *fichier* passé en paramètre.

Dans le répertoire source, nous avec aussi un fichier pour chaque conversion (ppm vers pgm et ppm vers pbm). Ces fonctions se ressembles beaucoup, il y a juste le calcul pour la valeur du pixel qui change. Le fonctionnement commun est simple, nous sauvegardons d'abords les informations dans des nouvelles structures (notamment la hauteur, la largeur, le type…). Puis, pour chaque pixel on fait les opérations sur les couleurs. Pour acceder aux différentes couleurs, nous utilisons des masques et des décalage qui sont stockés dans des *#define*.

Nous avons aussi créé un fichier *ecrire_fichier.c* qui contient la fonction `int write_file(struct structimg p, char* nom_fichier)` et qui nous permet simplement de créer un fichier pour sauvegarder l'image contenue dans la structure passée en paramètre. Il y a donc deux chemins pour l'extension (nuance de gris ou noir & blanc) puis on écrit toutes les donnés nécessaires pour remplir le fichier.

Enfin il y a le fichier *main.c* qui exécute les commandes une par une selon les paramètres.

Pour finir, nous avons créer un fichier Makefile qui permet de compiler tout le programme et de créer un executable sous *Application/bin*.

Mode d'emploi

Ce programme est très simple à utiliser : après avoir fait la commande *make* dans le dossier *Application/the_sources/* il faut exécuter le l'application dans le dossier *Application/bin* avec la commande *./main [OPTIONS]*. Il est nécessaire d'avoir au moins une des deux options suivantes dans la commande : -g pour créer un fichier .pgm ou -b pour créer un fichier .pbm.

Optionnellement : Il faut rajouter le chemin d'une image que l'on veut transformer. Si il n'y a pas de chemin spécifié, l'entrée standard (*stdin*) servira de fichier. Pour utiliser l'entrée standard, il respecter le même format qu'un fichier, et finir en appuyant deux fois sur les touches CTRL+D (la première fois permet de finir la lecture du flux, et la seconde indique la fin du fichier).

La ou les image(s) crée(s) sera/seront sauvegardée(s) dans le même répertoire que l'image au format .ppm, ou dans le fichier */Exemples/image.p(g/b)m*.

Exemples

Compilation du programme :

```
mars@pc:~/polytech/S2/CPS/Decamps_Lagrange$ cd Application/the_sources/
mars@pc:~/polytech/S2/CPS/Decamps_Lagrange/Application/the_sources$ make
gcc -o main.o -c main.c -I ../the_headers
gcc -o P2.o -c P2.c -I ../the_headers
gcc -o P1.o -c P1.c -I ../the_headers
gcc -o util.o -c util.c -I ../the_headers
gcc -o ecrire_fichier.o -c ecrire_fichier.c -I ../the_headers
gcc main.o P2.o P1.o util.o ecrire_fichier.o -o ../bin/main -lm
rm -f *.o *~
mars@pc:~/polytech/S2/CPS/Decamps_Lagrange/Application/the_sources$
```

Exécution sur une image dans Exemples/ :

```
mars@pc:~/polytech/S2/CPS/Decamps_Lagrange$ Application/bin/main -b Exemples/cha
let.ppm -g
On sauve l'image au format PGM dans le fichier Exemples/chalet.pgm
On sauve l'image au format PBM dans le fichier Exemples/chalet.pbm
mars@pc:~/polytech/S2/CPS/Decamps_Lagrange$
```

Exécution sur l'entrée standard, on peut voir ici un bug au niveau du format… Non résolu :

```
mars@pc:~/polytech/S2/CPS/Decamps_Lagrange/Application/bin$ ./main -b -g
P3
3 2
255
255 0 0 0 255 0 0 0 255
255 255 0       255 255 255     0 0 0
On sauve l'image au format PGM dans le fichier ../../Exemples/image.pgm1
On sauve l'image au format PBM dans le fichier ../../Exemples/image.pbm
```

Exécution avec un pipe, résultats avec des extensions modifiée… :

```
mars@pc:~/polytech/S2/CPS/Decamps_Lagrange/Application/bin$ cat ../../Exemples/i
mg.ppm | ./main -g -b
On sauve l'image au format PGM dans le fichier ../../Exemples/image.pgm1
On sauve l'image au format PBM dans le fichier ../../Exemples/image.pbm
```

Limitations

Nous n'avons pas réussi à faire fonctionner correctement l'application avec un pipe et sur l'entrée standard. Le bug est le suivant : Si nous exécutons la commande « cat ../../Exemples/img.ppm | ../bin/main -g -b » dans le dossier the_sources, l'application va bien lire l'entrée standard, faire les bonnes modifications, mais au moment de la sauvegarde, nous obtenons un fichier avec une extension .p(b/g)m1#. Après un débuguage, nous nous somme rendu compte que cette modification d'extension se fait lors de l'ouverture du fichier pour la sauvegarde (ligne 20 ou 29 de *ecrire_fichier.c*). Nous avons pourtant bien placé un caractère '\0' par sûreté mais rien n'y fait, de plus on peut voir sur la troisième capture d'écran ci-dessus que le bug est « aléatoire ». Nous ne comprenons vraiment pas d'où cela vient. L'image reste tout de même visible et peut être interprétée par un lecteur d'image.