TP/TD 9 : Entrées/Sorties binaires

1 Le format BMP

BMP, ou "bitmap", est comme son nom l'indique un format d'image matricielle. L'image est considérée comme une "carte de points", et le fichier .bmp contient la définition de chacun des pixels de l'image. Chaque pixel est représenté par sa couleur, où plutôt par sa proportion de chacune des trois couleurs fondamentales : Rouge, Vert et Bleu.

En plus des pixels de l'image, un fichier BMP contient toujours un en-tête (header) comportant quelques informations importantes comme la taille de l'image, sa largeur, sa hauteur, le type de compression, etc... De façon facultative, cet en-tête peut être suivi d'une palette spécifiant le champ des couleurs utilisées.

Nous nous intéressons dans ce TP au format BMP 24 bits non compressé. Dans celui-ci, chaque pixel est codé sur 24 bits, soit 3 octets, un par couleur fondamentale : $Pixel = (R, V, B) \times 8bits = 24bits$. Le taux de chacune des couleurs (R,V,B) est donc exprimé entre 0 et 255, toutes les couleurs sont représentées et la partie palette du fichier est inutile. L'en-tête dans ce format a une taille de 54 octets et les données qu'il contient sont les suivantes (et dans cet ordre) :

Nom	Taille en octets	Signification
Identifier	2	Contient toujours l'octet 'B' suivit de l'octet 'M'
FileSize	4	Taille totale du fichier en octets
Reserved	4	Champ réservé, doit être égal à 0
DataOffset	4	Nb d'octets séparant le début du fichier des données de l'image
HeaderSize	4	Taille en octets de l'en-tête
Width	4	Largeur de l'image en pixels
Height	4	Hauteur de l'image en pixels
Planes	2	Nombre de plans
BitsPerPixels	2	Nb de bits nécessaires pour représenter un pixel
Compression	4	Type de compression (0 = non compressé)
BitmapDataSize	4	Taille en octets des données de l'image
HResolution	4	Résolution horizontale de l'image en pixels par mètre
VResolution	4	Résolution verticale de l'image en pixels par mètre
Colors	4	Nombre de couleurs dans l'image
ImportantColors	4	Nombre de couleurs importantes

Question 1

Vous allez écrire la fonction *loadBMPImage* qui prend en paramètre le nom d'un fichier contenant une image BMP. Avant toute chose, cette fonction doit vérifier que le fichier fourni respecte bien le format BMP 24 bits non compressé. Pour cela, vous devez ouvrir le fichier puis lire et analyser certaines valeurs de l'en-tête. Vous devez en particulier vérifier que les 2 premiers octets sont bien 'B' et 'M'. Si c'est le cas, vous pouvez également vérifier que le format est bien 24 bits non compressé, et que la taille du fichier est bien cohérente avec la taille (hauteur×largeur) de l'image.

Notes : Récupérez le code à compléter et l'image de test fournis avec le sujet. Dans tout ce TP, vous devez utilisez des appels systèmes pour effectuer les opérations de lecture/écriture binaires. Le chapitre 2 du manuel vous sera donc utile, et en particulier les fonctions *open, read, lseek*.

Question 2

Maintenant que l'on sait qu'on a une image au format BMP, la fonction *loadBMPImage* va instancier une variable de type *BMPImage* à partir du fichier fourni.
Complétez la structure *BMPImage* avec les variables de l'en-tête qui vous semble pertinentes pour les traitements futurs de l'image. Faites le nécessaire pour remplir ces champs dans la fonction *loadBMPImage*.

Question 3

Au delà de l'en-tête, nous avons besoin de récupérer le contenu de l'image : les pixels. Complétez la structure *Pixel*. Créez le champ *pixels* dans la structure *BMPImage* qui contiendra la matrice de pixels. Faites le nécessaire pour allouer et remplir cette matrice dans la suite de la fonction *loadBMPImage*.

2 Cache-cache numérique

La stéganographie est l'art de dissimuler de l'information dans un contenu sans modifier l'apparence de ce dernier. L'oeil humain n'est pas très performant pour déceler les nuances de couleurs très proches. Nous pouvons donc profiter de cela pour dissimuler une image à l'intérieur d'une autre image. La technique que nous allons utiliser s'appelle LSB pour "Least Significant Bits". Comme son nom l'indique, cela consiste à remplacer les bits de poids faible de l'image "hôte" par les bits de poids fort de l'image à dissimuler. Ainsi, les couleurs de l'image envoyée vont être modifiées, de façon peu perceptible à l'oeil nu. Mais si le recepteur de l'image sait qu'une autre image est en fait cachée dans celle ci, il va pouvoir la décoder, dans une version un peu dégradée.

L'image *cat.bmp* fournie avec le sujet est en apparence une simple photo de chat comme on en trouve beaucoup sur internet. En réalité, elle dissimule une autre photo. Dans les questions suivantes, vous allez écrire le code nécessaire pour découvrir l'image cachée derrière cet innocent chat.

Question 1

Complètez la fonction *extractHiddenPixel* qui prend en paramètre un pixel (de l'image hôte) et va retourner un nouveau pixel (de l'image dissimulée) créé à partir des bits de poids faibles du pixel d'entrée. La technique qui a été utilisée est un entralecement 5 bits / 3 bits.

Notes : Vous allez devoir utiliser les opérations de calcul bit-à-bit comme &, | et les décalages : >> et <<. Pour débugguer, l'affichage d'un nombre en hexadécimal peut se faire avec *printf* avec le format "%02*hhX*" (ex: 00011010 sera affiché 1*A*).

Question 2

Une fois que cela fonctionne, vous pouvez complètez la fonction *extractHiddenImage* qui va créer une nouvelle BMPImage qui correspond à l'image extraite.

Question 3

Afin de pouvoir visualiser le résultat, complètez maintenant la fonction *writeImage* qui va créer et écrire un fichier BMP. Ce fichier aura le même en-tête que l'image d'entrée, et sa matrice de pixels sera celle de la BMPImage que vous venez de créer. La fonction *writeImage* prend donc en premier paramètre le fichier d'entrée qui sera lu uniquement pour récupérer le contenu de l'en-tête, pour le recopier dans le fichier nouvellement créé. L'ouverture de l'image découverte sera faite en dehors de votre programme C, avec l'outil de votre choix. Tada !

Question 4

Maintenant vous pouvez coder la partie dissimulation en complétant les fonctions *hidePixel* et *hideImage*.

Notes : Attention, les deux images doivent avoir la même taille. Vous pouvez facilement convertir des images au format BMP avec le logiciel *GIMP* par exemple, en sélectionnant bien le format 24 bits sans palette de couleurs.