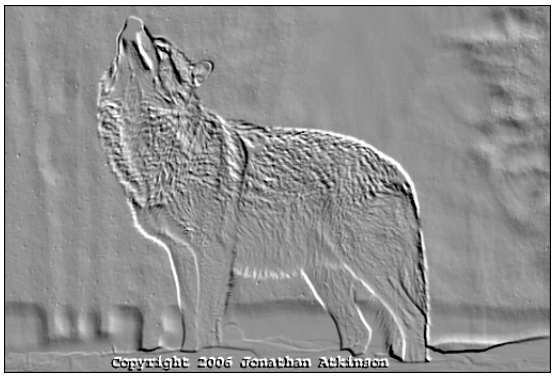
TP n° 3 : Tableaux et structures

NOM : ..... Prénom : .....

Attention, ce TP est noté :

- Les documents autorisés sont les diapositives de cours éventuellement annotées, les notes de TD et les sujets éventuellement annotés des TP 1 et 2.
- L'utilisation des logiciels de chat et d'e-mail est interdite. Les plagiats seront sanctionnés par un zéro.
- La note sera divisée par deux si le programme ne compile pas. Moralité : il faut absolument essayer de compiler au fur et à mesure ! Tout mettre en commentaires au dernier moment n'est pas une solution : les portions de code en commentaires ne seront pas évaluées.
- Vous avez cependant droit à deux « jokers » au cours des trois heures : vous pouvez appeler deux fois l'encadrant pour une erreur de compilation. Vous devrez par contre trouver vous-même les calculs à effectuer.
- Le rendu du code se fait sur Spiral, dans l'espace « Echange de docs » correspondant à votre groupe. Attention de bien uploader les fichiers .c et non .c~. Cet espace Spiral sera automatiquement fermé à 19h, réservez impérativement les 10 dernières minutes à cela. Les codes rendus en retard par e-mail ne seront pas évalués.

L'objectif de ce TP est de programmer différents traitements sur des images au format BMP 24bits : vous allez programmer le passage d'une image en niveaux de gris, puis un filtre d'embossage. Les procédures de lecture et d'écriture de fichier vous sont fournies, de sorte à ce que vous vous concentriez sur les calculs des valeurs RGB des pixels.



En haut à gauche : image originale, en couleurs

En haut à droite : image en niveaux de gris

Ci-contre : image en niveaux de gris après traitement par un filtre d'embossage, donnant une impression de relief.

Rappel : Bitmap, connu aussi sous son abréviation BMP, est un format d'image matricielle ouvert développé par Microsoft et IBM. C'est l'un des formats d'images les plus simples à développer et à utiliser pour programmer. Il est lisible par quasiment tous les visualiseurs et éditeurs d'images. Le fichier se découpe en 2 zones :

- L'en-tête du fichier : on y trouve notamment la largeur et la hauteur de l'image, en nombre de pixels.
- Les données relatives à l'image : les pixels de l'image sont codés ligne par ligne, en partant de la ligne inférieure de l'image. Si l'image est codée en 24 bits (ce qui sera le cas pour tout ce TP), chaque pixel est codé par trois octets (trois unsigned char) codant successivement les niveaux de bleu, vert et rouge. Chacun de ces trois niveaux est compris entre 0 et 255. Exemples de pixels :
  (0, 0, 0) → noir
  (255, 255, 255) → blanc
  (255, 0, 0) → rouge
  (0, 255, 0) → vert
  (0, 0, 255) → bleu
  (255, 192, 203) → une nuance possible de rose

Commencez par créer un répertoire TP3 à l'intérieur de votre répertoire LIF5, en utilisant les lignes de commandes vues au TP1 (cd, mkdir, ls…), et placez-y les fichiers disponibles sur Spiral :

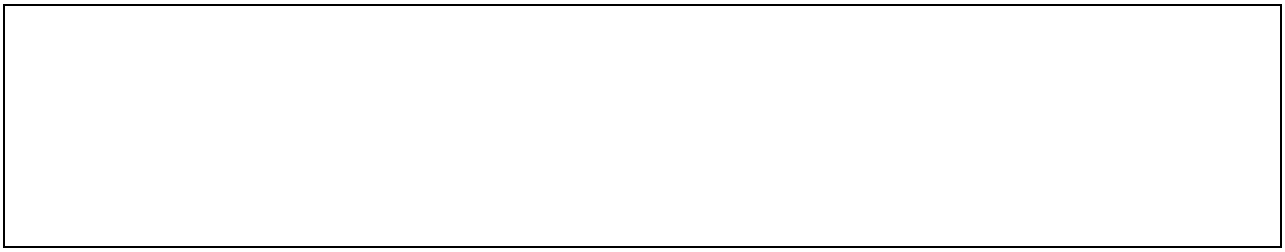
- tp3-lundi.c : squelette de code à compléter,
- grey_wolf.bmp : image originale,
- verif-grayscale.bmp : ce que votre traitement « niveaux de gris » doit produire,
- verif-embossage.bmp : ce que votre traitement « embossage » doit produire.

Exercice 1 : Lire l'image originale (chargement en mémoire vive)

Le fichier tp3-lundi.c définit le type pixel, sous forme d'une struct. Il contient aussi deux procédures utiles pour lire un fichier BMP : lireEntete et remplirTableauPixelsDepuisFichier.

Lisez attentivement les entêtes de ces procédures, puis complétez le « main » pour récupérer la largeur et la hauteur de l'image originale (grey_wolf.bmp), allouer un tableau 1D de taille suffisante pour contenir tous les pixels, et remplir ce tableau à partir du fichier. Vérifiez que votre code ne contient pas d'erreur de compilation ni d'exécution.

Pour une image de largeur L et de hauteur H, quelle instruction faudrait-il écrire pour afficher les 3 niveaux R, G, B du pixel qui se trouve à la 3è ligne (en partant du bas), et à la 9è colonne ?



Exercice 2 : Traitement « niveaux de gris »

Dans le codage RGB, si l'on fixe les trois octets à la même valeur (par exemple 125, 125, 125), on obtient une nuance de gris. La conversion d'une image en niveaux de gris consiste à calculer pour chaque pixel la moyenne de ses trois niveaux R, G, B : dans l'image convertie, le pixel aura ses trois niveaux R, G, B égaux à cette moyenne.

Ecrivez au-dessus du main la procédure traitementNiveauxDeGris, qui prend en paramètres un tableau de pixels correspondant à l'image originale (mode donnée), un tableau de pixels assez grand pour contenir l'image convertie (mode résultat), ainsi que la largeur et la hauteur de l'image (mode donnée). Cette procédure remplit le second tableau de pixels selon l'algorithme décrit ci-dessus. Veillez à indiquer les pré- et post-conditions en commentaires.

Complétez le main pour qu'il appelle correctement cette procédure, puis écrive le tableau de pixels résultant dans un fichier au format BMP que vous appellerez « grayscale.bmp » (utilisez pour cela la procédure ecrireFichier). Vérifiez que votre code ne contient pas d'erreur de compilation ni d'exécution. Vérifiez que l'image produite correspond au résultat attendu.

Exercice 3 : Traitement « embossage »

Le filtre d'embossage donne l'illusion optique que certains objets de l'image sont plus ou moins près du fond, ce qui crée un effet de relief. Cet effet est obtenu de la façon suivante :

- on part d'une image en niveaux de gris,
- pour chaque pixel de l'image de sortie (sauf ceux en bordure de l'image), on calcule le niveau de gris en tenant compte du niveau de ce pixel et de 2 de ses voisins dans l'image originale, selon le calcul suivant :

			2	
		−1		
	−1			

Niveau brut pour le pixel central = 2*niveau du pixel voisin nord-est
- niveau de ce pixel
- niveau du pixel voisin sud-ouest
+ 128

Si ce niveau brut excède 255, on met le pixel à 255.
S'il est négatif, on met le pixel à zéro.

- ce calcul n'est pas réalisable pour les pixels en bordure de l'image, on les mettra donc en noir.

Est-il pertinent d'utiliser une variable de type « unsigned char » pour stocker le niveau brut ? Pourquoi ? Quel type suggérez-vous ?

Ecrivez au-dessus du main la procédure traitementEmbossage, qui prend en paramètres un tableau de pixels correspondant à une image en niveaux de gris (mode donnée), un tableau de pixels assez grand pour contenir l'image embossée (mode résultat), ainsi que la largeur et la hauteur de l'image (mode donnée). Cette procédure remplit le second tableau de pixels selon l'algorithme décrit ci-dessus. Veillez à indiquer les pré- et post-conditions en commentaires.

Complétez le main pour qu'il appelle correctement cette procédure, puis écrive le tableau de pixels résultant dans un fichier au format BMP que vous appellerez « embossage.bmp ». Vérifiez que votre code ne contient pas d'erreur de compilation ni d'exécution. Vérifiez que l'image produite correspond au résultat attendu.

Votre main doit se terminer proprement et donc libérer la mémoire éventuellement allouée dynamiquement.