Codage
-
Images

Emmanuel Jeandel (emmanuel.jeandel@lif.univ-mrs.fr)
`http://www.lif.univ-mrs.fr/~ejeandel/enseignement.html`

15 février 2011

Formats PPM et PGM

PPM et PGM sont des formats de fichiers graphiques très faciles d'usage. On pourra exécuter les commandes `man pgm` et `man ppm` pour plus d'informations.

PGM Un fichier PGM est un format d'image en tons de gris, et est de la forme suivante
- Une entête consistant en la chaîne `P5`
- Sur la ligne suivante, longueur et hauteur (en décimal)
- Sur la ligne suivante, le nombre maximal de couleurs (toujours 255 dans le TD)
- Dès la ligne suivante, tous les pixels de l'image : Chaque caractère correspond à la "couleur" d'un pixel.

PPM Un fichier PPM est un format d'image couleur et est de la forme suivante
- Une entête consistant en la chaîne `P6`
- Sur la ligne suivante, longueur et hauteur (en décimal)
- Sur la ligne suivante, le nombre maximal de niveaux de couleurs (toujours 255 dans le TD)
- Dès la ligne suivante, tous les pixels de l'image : Chaque triplet caractère correspond à l'intensité de rouge, vert, bleu du pixel.

Dans PPM comme dans PGM, les valeurs d'intensité sont nonlinéaires, et proportionnelles à l'intensité de chaque couleur dans le pixel, avec une correction γ de 2.2. L'illuminant est $D65$.

Vous pouvez trouver sur la page du cours deux images, l'une en format PPM l'autre en format PGM.

Q 1) Donner un algorithme qui convertit une image couleur en niveaux de gris. Pour les détails précis, on se rapportera à la page de manuel de `ppmtopgm`.

Réduction de couleurs

Supposons avoir une image avec beaucoup de couleurs, et vouloir réduire le nombre de couleurs. Ceci s'effectue en deux étapes : d'abord sélectionner les couleurs qu'on prend puis transformer l'image pour qu'elle ait moins de couleurs Le choix des nouvelles couleurs n'est pas toujours à faire ; elles peuvent être imposées.

Réduction des couleurs

Q 2) Proposer un mécanisme de sélection de couleurs dans une image.

La méthode Median-Cut procède, pour les images en tons de gris, de la façon suivante : On commence par mettre tous les pixels dans une boite. Ensuite, à chaque étape, on coupe chaque boîte en deux en mettant les couleurs d'intensité les plus faibles dans une même boite, et les autres ensemble, de façon à ce qu'il y ait autant de pixels dans chaque boite.

Q 3) Effectuer l'algorithme lorsque les couleurs sont les suivantes. On veut obtenir à la fin 4 couleurs.

Couleur	Nb de pixels
0	1
1	5
2	2
10	15
30	18
37	3
63	7

Pour une image en couleur, l'algorithme choisit à chaque étape, pour diviser une boîte, la couleur pour laquelle la variation d'intensité (ou de luminosité) entre le minimum et le maximum est la plus importante.

Q 4) Effectuer l'algorithme (deux fois, une pour luminosité et une pour intensité) lorsque les couleurs sont les suivantes. On veut obtenir à la fin 6 couleurs.

R	V	B	Nb de pixel
32	28	15	60
84	51	22	44
101	88	29	56
80	84	79	27
44	110	207	48
79	107	158	15
88	156	222	48
61	137	213	27
145	103	34	48
150	108	74	35
165	149	67	35
158	178	211	43
179	200	226	25
213	216	223	34
190	186	160	31
116	137	56	31

Une fois découpé en boîtes, il faut choisir, pour chaque boîte, le pixel qui la représente le mieux.

Q 5) Proposer une méthode.

Transformation de l'image

On suppose maintenant connu les couleurs à utiliser, il reste à expliquer comment transformer l'image pour qu'elle ne contienne que ces couleurs.

Q 6) Décrire une méthode.

On suppose que l'image de départ est en ton de gris, et qu'on veut la mettre en deux couleurs (noir et blanc)

Q 7) Quel est le défaut de votre méthode ? Examiner ce qui se passe sur une image composée de deux bandes de deux gris différents.

Q 8) On décide d'ajouter un bruit aléatoire à chaque pixel : on tire pour chaque i et j un entier c entre -64 et 64 et on transforme $T[i][j]$ en $T[i][j] + c$. On utilise ensuite la technique précédente pour convertir. Expliquer pourquoi le résultat aura l'air meilleur.

La méthode de Floyd-Steinberg est une variante sur ce principe de bruit aléatoire, très adaptée aux images en niveaux de gris (lorsqu'il y a des couleurs, on travaille séparemment sur chaque couleur). En voilà le principe. On change la couleur de chaque pixel de haut en bas et de gauche à droite. Supposons qu'on remplace la couleur C du pixel (x, y) par la couleur C_1. On commet donc une erreur ϵ. L'idée est alors de modifier les pixels aux alentours de (x, y) qui n'ont pas encore été transformés pour tenir compte de cette erreur : On ajoute donc $7/16$ de l'erreur ϵ sur le pixel de droite, $3/16$ sur le pixel en bas à gauche, $5/16$ sur le pixel en bas, $1/16$ sur le pixel en bas à droite.

On peut visualiser l'opération ainsi :

	$\times$	$7/16\epsilon$
$3/16\epsilon$	$5/16\epsilon$	$1/16\epsilon$

Partons par exemple de l'image suivante, qu'on veut mettre en noir et blanc : on va donc transformer tout pixel en 0 ou 255 en suivant la méthod de Floyd-Steinberg.

162	149	33	243	142
95	126	240	125	174
160	37	49	58	239

On transforme le 162 en 255. Ce faisant, on commet une erreur de $\epsilon = 162 - 255 = -93$. On va donc ajouter $7/16\epsilon = -40$ au pixel à sa droite, ce qui fait $149 - 40 = 109$ et ainsi de suite pour les autres pixels alentour (à noter que le $3/16\epsilon$ du pixel en bas à gauche est perdu puisqu'on est sur le bord de l'image)

255	109	33	243	142
66	120	240	125	174
160	37	49	58	239

On remplace ensuite le 109 par 0 commettant alors une erreur de $\epsilon = 109$, on corrige donc le pixel de droite par $7/16\epsilon = 48$ qui devient donc $33 + 48 = 81$ et on corrige aussi les autres pixels alentour :

255	0	81	243	142
86	154	246	125	174
160	37	49	58	239

et ainsi de suite jusqu'à ce que tout le tableau soit rempli de 0 et de 255.

Q 9) Écrire l'algorithme. On essaiera d'avoir en permanence que deux lignes au maximum de l'image en mémoire.

Une idée d'algorithme est la suivante :

- Lire la première ligne, la mettre dans un tableau a
- Lire la deuxième ligne, qu'on met dans un tableau b
- Traiter le tableau a (ce qui devrait modifier le tableau b)
- Renvoyer le résultat du traitement du tableau a
- Recopier le tableau b dans le tableau a
- Lire une nouvelle ligne ligne, la mettre dans b
- Recommencer à la troisième étape.

Attention au cas particulier de la dernière ligne.

TP

Q 10) Ecrire un logiciel qui convertir une image en couleurs au format PPM en une image en niveaux de gris au format PGM.

Q 11) Ecrire un logiciel qui convertir une image en niveaux de gris en une image en noir en blanc, les deux au format PGM. On utilisera l'algorithme de Floyd-Steinberg.

Quelques indications Pour lire le fichier, on pourra par exemple utiliser le code suivant

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
   int w,h,i;
   unsigned int c;
   FILE* f = fopen("two.ppm", "r");
   fscanf(f, "P6\n");
   fscanf(f, "%d %d\n",&w, &h);
   fscanf(f, "255\n");
   for (i = 0; i < w*h*3; i++)
 {
   c = fgetc(f);
 }
   fclose(f);
}
```