Projet Maths pour l'info

—IMAC 1—

Mosaique

L'ouverture d'un appareil photo est limitée et il arrive qu'elle ne soit pas suffisante pour la prise de vue désirée. Tout le monde n'a pas un appareil pouvant faire des photos panoramiques. Ce projet propose une fa̧on de combiner plusieurs images prises à partir du même point de vue pour en faire un panoramique.

1 Position du problème

Si l'on dispose de plusieurs images comportant chacune des objets communs, il est possible de les regrouper pour ne former plus qu'une seule grande image. Il faut toutefois que ces images aient été prises du même point de vue.

3 images prises à partir du même endroit.

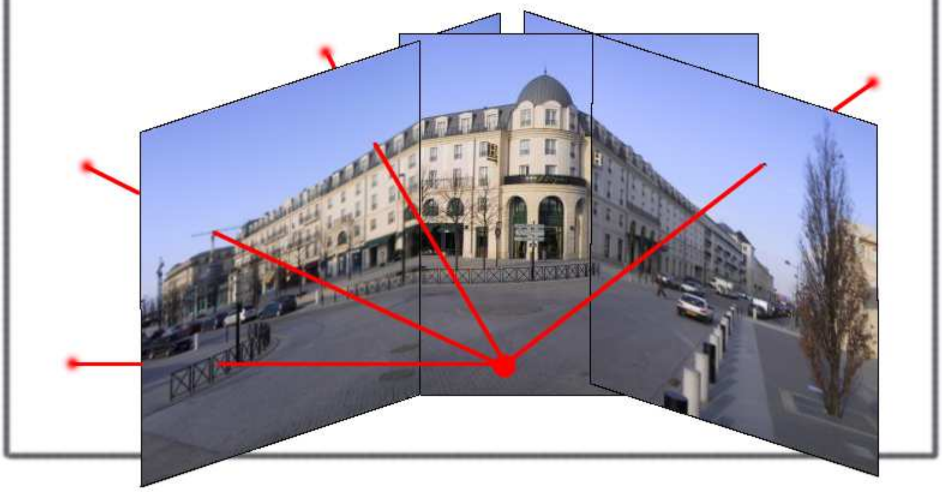
On peut penser qu'il suffit de déplacer les images (translation, rotation), voire même de les redimensioner pour obtenir un panoramique. En fait, ce n'est pas suffisant.

Observer la cassure de la ligne de fuite
au niveau du toit gauche.

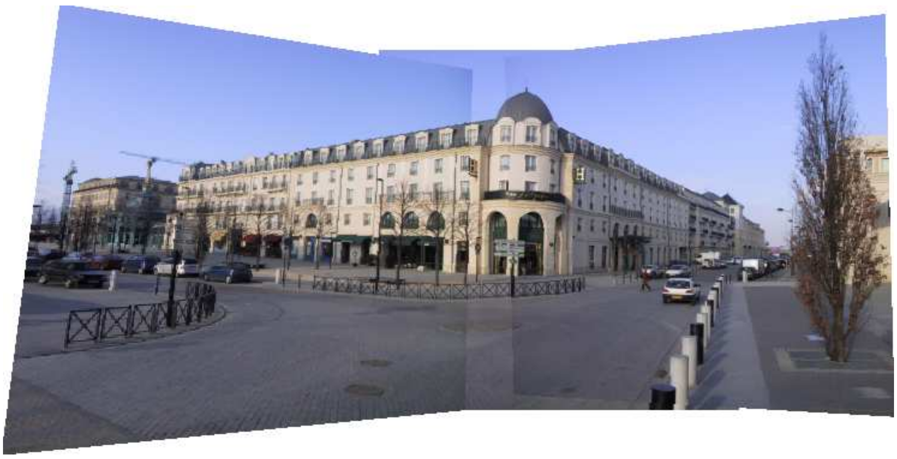
2 Méthodes de résolution

2.1 Principe

On voudrait projeter toutes les images sur le même plan. Pour des raisons pratiques, on choisit comme plan de projection le plan contenant l'image du milieu. Il ne reste plus qu'à projeter les autres images sur ce plan.



La projection à appliquer à chacune des images des bords s'appelle une homographie (parfois aussi appelée colinéation ou transformation projective). Sur l'image suivante, les images de droite et de gauche ont été projetées dans dans le plan contenant l'image centrale.



Une telle transformation ne conserve ni les longueurs, ni les angles et peut s'écrire sous forme matricielle:

$$\begin{pmatrix} x' \\ y' \\ w' \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$\text{soit} \quad \mathbf{x}' = H\mathbf{x}$$

$\mathbf{x} = (x, y, 1)^\top$ et $\mathbf{x}' = (x', y', w')^\top = \left(\frac{x'}{w'}, \frac{y'}{w'}, 1\right)^\top$ sont les coordonnées homogènes des pixels de l'image de départ et de l'image déformée. Pour déformer une image, il suffit donc d'appliquer une homographie à tous ses pixels.

Redressement d'une image par homographie.

2.2 Calcul d'une homographie

Une homographie est une matrice 3×3 et possède donc 9 éléments indépendants. Cette matrice est utilisée en coordonnées homogènes, elle est donc invariante par changement d'échelle et on peut alors la multiplier par une constante réelle non nulle sans changer ses effets sur une image. Une homographie a donc 8 degrés de liberté et elle est déterminée par au minimum 4 correspondances de points (qui ont chacuns 2 degrés de liberté). Pour calculer une homographie, il suffit de calculer la matrice H telle que $\mathbf{x}'_i = H\mathbf{x}_i$ pour chaque couple de points $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ où les $\mathbf{x}_i$ de l'image de départ correspondent aux $\mathbf{x}'_i$ de l'image d'arrivée.

Pour résoudre $\mathbf{x}'_i = H\mathbf{x}_i$, il vaut mieux partir de $\mathbf{x}'_i \wedge H\mathbf{x}_i = \mathbf{0}$ ('$\wedge$' correspond au produit vectoriel). En séparant les lignes de H, on obtient :

$$H = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} = \begin{pmatrix} \mathbf{h}_1^\top \\ \mathbf{h}_2^\top \\ \mathbf{h}_3^\top \end{pmatrix}$$

soit

$$\mathbf{h}_1 = \begin{pmatrix} h_{11} \\ h_{12} \\ h_{13} \end{pmatrix} \quad \mathbf{h}_2 = \begin{pmatrix} h_{21} \\ h_{22} \\ h_{23} \end{pmatrix} \quad \mathbf{h}_3 = \begin{pmatrix} h_{31} \\ h_{32} \\ h_{33} \end{pmatrix}$$

On obtient alors

$$H\mathbf{x}_i = \begin{pmatrix} \mathbf{h}_1^\top.\mathbf{x}_i \\ \mathbf{h}_2^\top.\mathbf{x}_i \\ \mathbf{h}_3^\top.\mathbf{x}_i \end{pmatrix}$$

et

$$\begin{pmatrix} x'_i \\ y'_i \\ w'_i \end{pmatrix} \wedge \begin{pmatrix} \mathbf{h}_1^\top.\mathbf{x}_i \\ \mathbf{h}_2^\top.\mathbf{x}_i \\ \mathbf{h}_3^\top.\mathbf{x}_i \end{pmatrix} = \begin{pmatrix} y'_i.\mathbf{h}_3^\top.\mathbf{x}_i - w'_i.\mathbf{h}_2^\top.\mathbf{x}_i \\ w'_i.\mathbf{h}_1^\top.\mathbf{x}_i - x'_i.\mathbf{h}_3^\top.\mathbf{x}_i \\ x'_i.\mathbf{h}_2^\top.\mathbf{x}_i - y'_i.\mathbf{h}_1^\top.\mathbf{x}_i \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

ce qui donne

$$\begin{pmatrix} 0\ 0\ 0 & -w'_i\mathbf{x}_i^\top & y'_i\mathbf{x}_i^\top \\ w'_i\mathbf{x}_i^\top & 0\ 0\ 0 & -x'_i\mathbf{x}_i^\top \\ -y'_i\mathbf{x}_i^\top & x'_i\mathbf{x}_i^\top & 0\ 0\ 0 \end{pmatrix} \begin{pmatrix} \mathbf{h}_1^\top \\ \mathbf{h}_2^\top \\ \mathbf{h}_3^\top \end{pmatrix} = \begin{pmatrix} 0\ 0\ 0 & -w'_i\mathbf{x}_i^\top & y'_i\mathbf{x}_i^\top \\ w'_i\mathbf{x}_i^\top & 0\ 0\ 0 & -x'_i\mathbf{x}_i^\top \\ -y'_i\mathbf{x}_i^\top & x'_i\mathbf{x}_i^\top & 0\ 0\ 0 \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

ce qui correspond à un sytème d'équations linéaires $A\mathbf{h} = \mathbf{0}$ avec

$$A = \begin{pmatrix} 0\ 0\ 0 & -w'_i\mathbf{x}_i^\top & y'_i\mathbf{x}_i^\top \\ w'_i\mathbf{x}_i^\top & 0\ 0\ 0 & -x'_i\mathbf{x}_i^\top \\ -y'_i\mathbf{x}_i^\top & x'_i\mathbf{x}_i^\top & 0\ 0\ 0 \end{pmatrix} \ \ et \ \ \mathbf{h} = \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{pmatrix}$$

Il se trouve que la troisième équation est une combinaison linéaire des deux premières, on peut donc s'en passer. On obtient alors pour le couple $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ le système suivant :

$$\begin{pmatrix} 0\ 0\ 0 & -w'_i\mathbf{x}_i^\top & y'_i\mathbf{x}_i^\top \\ w'_i\mathbf{x}_i^\top & 0\ 0\ 0 & -x'_i\mathbf{x}_i^\top \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

En combinant ce système pour les 4 points de correspondance nécessaires, on obtient le système à 8 équations et 9 inconnues suivant :

$$\begin{pmatrix} 0\,0\,0 & -w'_1\mathbf{x}_1^\top & y'_1\mathbf{x}_1^\top \\ w'_1\mathbf{x}_1^\top & 0\,0\,0 & -x'_1\mathbf{x}_1^\top \\ 0\,0\,0 & -w'_2\mathbf{x}_2^\top & y'_2\mathbf{x}_2^\top \\ w'_2\mathbf{x}_2^\top & 0\,0\,0 & -x'_2\mathbf{x}_2^\top \\ 0\,0\,0 & -w'_3\mathbf{x}_3^\top & y'_3\mathbf{x}_3^\top \\ w'_3\mathbf{x}_3^\top & 0\,0\,0 & -x'_3\mathbf{x}_3^\top \\ 0\,0\,0 & -w'_4\mathbf{x}_4^\top & y'_4\mathbf{x}_4^\top \\ w'_4\mathbf{x}_4^\top & 0\,0\,0 & -x'_4\mathbf{x}_4^\top \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

soit en développant

$$\begin{pmatrix} 0 & 0 & 0 & -w'_1x_1 & -w'_1y_1 & -w'_1w_1 & y'_1x_1 & y'_1y_1 & y'_1w_1 \\ w'_1x_1 & w'_1y_1 & w'_1w_1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 & -x'_1w_1 \\ 0 & 0 & 0 & -w'_2x_2 & -w'_2y_2 & -w'_2w_2 & y'_2x_2 & y'_2y_2 & y'_2w_2 \\ w'_2x_2 & w'_2y_2 & w'_2w_2 & 0 & 0 & 0 & -x'_2x_2 & -x'_2y_2 & -x'_2w_2 \\ 0 & 0 & 0 & -w'_3x_3 & -w'_3y_3 & -w'_3w_3 & y'_3x_3 & y'_3y_3 & y'_3w_3 \\ w'_3x_3 & w'_3y_3 & w'_3w_3 & 0 & 0 & 0 & -x'_3x_3 & -x'_3y_3 & -x'_3w_3 \\ 0 & 0 & 0 & -w'_4x_4 & -w'_4y_4 & -w'_4w_4 & y'_4x_4 & y'_4y_4 & y'_4w_4 \\ w'_4x_4 & w'_4y_4 & w'_4w_4 & 0 & 0 & 0 & -x'_4x_4 & -x'_4y_4 & -x'_4w_4 \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Une solution évidente est $\mathbf{h} = \mathbf{0}$ mais elle n'est pas très intéressante. Pour éviter de trouver ce résultat, nous allons utiliser une astuce qui consiste à supposer h_9 non nul (ce qui est toujours le cas pour une homographie) et puisque H est invariant par facteur d'échelle, nous posons $h_9 = 1$. Le système à résoudre devient alors :

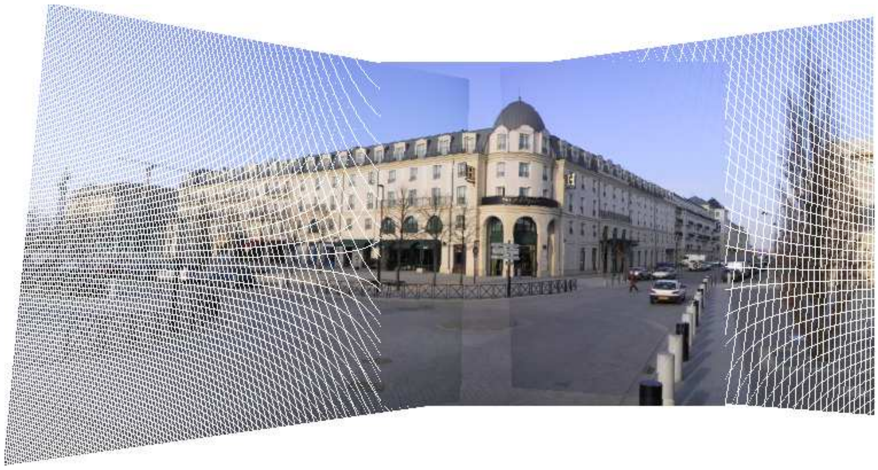
$$\begin{pmatrix} 0 & 0 & 0 & -w'_1x_1 & -w'_1y_1 & -w'_1w_1 & y'_1x_1 & y'_1y_1 \\ w'_1x_1 & w'_1y_1 & w'_1w_1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 \\ 0 & 0 & 0 & -w'_2x_2 & -w'_2y_2 & -w'_2w_2 & y'_2x_2 & y'_2y_2 \\ w'_2x_2 & w'_2y_2 & w'_2w_2 & 0 & 0 & 0 & -x'_2x_2 & -x'_2y_2 \\ 0 & 0 & 0 & -w'_3x_3 & -w'_3y_3 & -w'_3w_3 & y'_3x_3 & y'_3y_3 \\ w'_3x_3 & w'_3y_3 & w'_3w_3 & 0 & 0 & 0 & -x'_3x_3 & -x'_3y_3 \\ 0 & 0 & 0 & -w'_4x_4 & -w'_4y_4 & -w'_4w_4 & y'_4x_4 & y'_4y_4 \\ w'_4x_4 & w'_4y_4 & w'_4w_4 & 0 & 0 & 0 & -x'_4x_4 & -x'_4y_4 \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \end{pmatrix} = \begin{pmatrix} -y'_1w_1 \\ x'_1w_1 \\ -y'_2w_2 \\ x'_2w_2 \\ -y'_3w_3 \\ x'_3w_3 \\ -y'_4w_4 \\ x'_4w_4 \end{pmatrix}$$

C'est ce système qu'il faut résoudre et une fois $\mathbf{h}$ trouvé, on n'a plus qu'à faire :

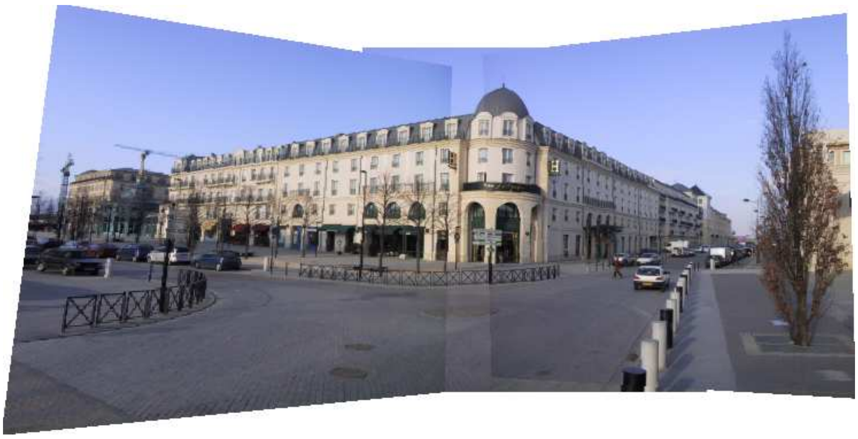
$$H = \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{pmatrix}$$

2.3 Utilisation d'une homographie

Une fois l'homographie H calculée, il nous suffit pour transformer une image d'appliquer cette homographie à chaque pixels de l'image. Cette méthode peut toutefois poser certains problèmes : l'image transformée n'occupe pas forcément tout l'espace de l'image d'arrivée. Comme on peut le voir sur l'image ci-dessous, certains points de l'image finale n'ont pas d'antécédents dans l'image de départ.



Une solution consiste à calculer $\mathbf{x}_i = H^{-1}\mathbf{x}'_i$ plutôt que $\mathbf{x}'_i = H\mathbf{x}_i$. Pour celà, après avoir inversé H, il suffit de parcourir l'image d'arrivée, de calculer $\mathbf{x}_i = H^{-1}\mathbf{x}'_i$ et de voir si le pixel $\mathbf{x}_i$ appartient à l'image de départ. S'il est effectivement dans l'image de départ, le pixel $\mathbf{x}'_i$ de l'image finale prend la même couleur que le pixel $\mathbf{x}_i$ de l'imade de départ. On obtient alors :



2.4 Systèmes surdéterminés

La méthode précédente fonctionne bien mais elle n'utilise que 4 point ce qui ne permet pas une très bonne précision. Pour être plus précis, il suffit de prendre plus de points mais le système à résoudre devient surdéterminé : il y a plus d'équations que d'inconnues et les équations supplémentaires ne sont pas nécessairement en accord avec les précédentes (elles ne sont pas forcément combinaisons linéaires des premières). Les solutions trouvées ne satisferont donc pas nécessairement toutes les équations mais tenteront d'en satisfaire un maximum (cf. méthode des moindres carrées).

Pour résoudre un système surdéterminé $A\mathbf{x} = \mathbf{b}$, on peut utiliser la matrice pseudo-inverse de A notée $A^\dagger$ avec $A^\dagger = (A^\top A)^{-1} A^\top$. Le résultat du système est alors $\mathbf{x} = A^\dagger \mathbf{b}$.

2.5 inversion d'une matrice

Pour inverser une matrice A, vous pouvez utiliser la propriété $AA^{-1} = Id$. Ainsi, pour trouver la j^{eme} colonne de A^{-1} que l'on notera $\mathbf{a}_j$, vous résoudrez $A\mathbf{a}_j = Id_j$ où Id_j est la j^{eme} colonne de la matrice identité Id.

3 Travail demandé

Le travail à effectuer sera réparti en 3 parties :

- la réalisation d'une bibliothèque mathématique en langage `C` permettant de gérer les matrices et les systèmes d'équations.
- la réalisation d'une bibliothèque image en langage `C` permettant de construire une mosaique à partir de 3 images.
- un rapport d'une centaine de pages.

3.1 bibliothèque mathématique

La bibliothèque mathématique doit permettre de résoudre un système d'équations linéaires par la méthode du pivot de Gauss (pivot total). Elle doit aussi pouvoir inverser une matrice en utilisant la méthode décrite au chapitre 2.5. Elle doit enfin pouvoir résoudre un système surdéterminé en utilisant la méthode du chapitre 2.4.

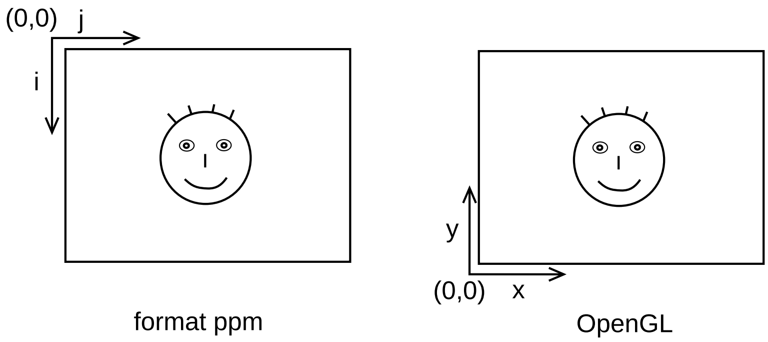
3.2 bibliothèque image

La bibliothèque image doit permettre de réaliser une mosaique à partir de 3 images. L'image du milieu (image2) servira d'image de référence. L'utilisateur devra selectionner à la souris un certain nombre de points de correspondances entre l'image de gauche (image1) et l'image centrale (image2) puis faire de même avec l'image de droite (image3) et l'image du milieu (image2). A partir de ces correspondances, votre programme devra calculer les matrices d'homographie H_{12} et H_{32} permettant de projeter l'image1 et l'image3 dans le repère de l'image2.

Vous utiliserez dans un premier temps l'homographie directe puis dans un second temps l'homographie inverse (cf chapitre 2.3). Votre programme sauvera l'image obtenue dans un fichier au format ppm.

Remarques :

- Pour connaitre la taille de l'image finale, il est recommandé d'appliquer H_{12} aux coins de l'image1 et H_{32} aux coins de l'image3 pour évaluer la taille des images transformées.
- Pour traiter les images dans votre programme, il est conseillé d'utiliser le même repère que celui d'OpenGL. Ce n'est pas le cas a priori :



3.3 rapport

Votre rapport devra comporter les points suivants :

- Les méthodes mathématiques qui ne figurent pas dans l'énoncé.
- Les algorithmes utilisés décrits de la façon la plus concise.
- Les difficultés rencontrées (d'ordre algorithmique).
- Des idées pour les problèmes non-résolus.
- Des idées d'améliorations.

3.4 informations générales

Vous commencerez par la partie mathématique et vous ne ferez pas la partie graphique tant que la première partie ne sera pas terminée.

Votre programme devra fonctionner sous Linux, vous fournirez un makefile et votre programme devra compiler sans warning. Un fichier d'aide sera lisible en exécutant votre programme avec l'option $-h$. Toutes les options doivent être gérées sur la ligne de commande.

4 Documentation technique

4.1 fichier.*ppm*

Il s'agit d'un format d'images reconnu par divers éditeurs d'images comme *gimp* ou *ee* (electric eyes). Il en existe plusieurs variantes : couleurs/niveaux de gris, ASCII/binaire. Le format le mieux adapté à notre problème est le format couleur en binaire. Le fichier commencera donc avec un en-tête correspondant.

Un pixel en couleur est défini par 3 nombres : un pour "la quantité" de rouge (R), un pour "la quantité" de vert (V) et un pour "la quantité" de bleu (B), noté RVB (RGB en anglais). Nous coderons ces quantités par des `unsigned char`. Un `unsigned char` est codé sur 8 bits, soit un octet. Ces nombres vont donc de 0 à 255.

Revenons au format *ppm*. Le premier nombre correspond au "numéro de variante" choisi, dans notre cas *P*6. Sur la ligne suivante, les deux nombres correspondent à la *largeur* et la *hauteur* de l'image. Enfin, le dernier nombre correspont à la valeur maximale que l'on puisse rencontrer (ici 255). Des commentaires peuvent être insérés dans l'en-tête, une ligne de commentaires commence par un #.
Chaque pixel sera écrit dans le fichier en procédant ligne par ligne, de gauche à doite, en commençant par le haut. Les pixels seront écrits sous forme de 3 octets écrits les uns à la suite des autres (RVB).

exemple:

```
#commentaires eventuels
P6
128 256
255
^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y
^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y
^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y
^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y
^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y^@y
^@y^@y^@y ...
```

5 Pour finir

Vous pouvez laisser libre cours à votre imagination, toute amélioration sera la bienvenue. Vous trouverez quelques informations complémentaires à l'adresse :

`http://www-igm.univ-mlv.fr/~vnozick/`

Bon courage.