



Projet Maths pour l'info

—IMAC 1—

Histogramme rééquilibré

Le traitement automatique d'images a de nombreuses applications, que ce soit dans le domaine de la photographie artistique ou dans celui de la photographie technique. Le problème proposé dans ce projet concerne la manipulation d'histogrammes permettant une meilleure lisibilité de l'image. Ce projet a pour but d'une part de vous entraîner à manipuler des concepts mathématiques et à les appliquer à des problèmes concrets. D'autre part, ce projet vous permet de vous familiariser avec les notions de traitement d'images. Enfin, ce projet est un bon entraînement à la programmation en langage C.

1 Position du problème

Lors d'une prise de vue, il peut arriver que la dynamique des couleurs soit mal répartie. Ce problème peut par exemple être lié à une sur-exposition, une sous-exposition, un contre-jour ou juste au mode de prise de vue. Le problème consiste ici à modifier les couleurs de l'image de telle sorte que le contraste soit augmenté sans créer de saturation. Cette fonctionnalité est disponible dans les logiciels de traitement d'images comme *Gimp* ou *Photoshop*. La figure 1 montre une application dans le domaine médical où l'aspect artistique est mis de côté au profit de la lisibilité de l'image. Il est fondamental pour une telle application que le traitement soit automatique compte tenu du nombre d'images à traiter dans certains cas.

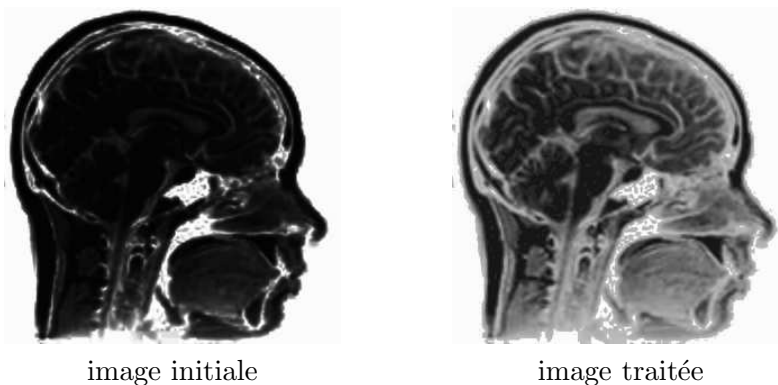


FIG. 1 – Exemple

la figure 2 montre une application plus ordinaire sur des images sous-exposées.

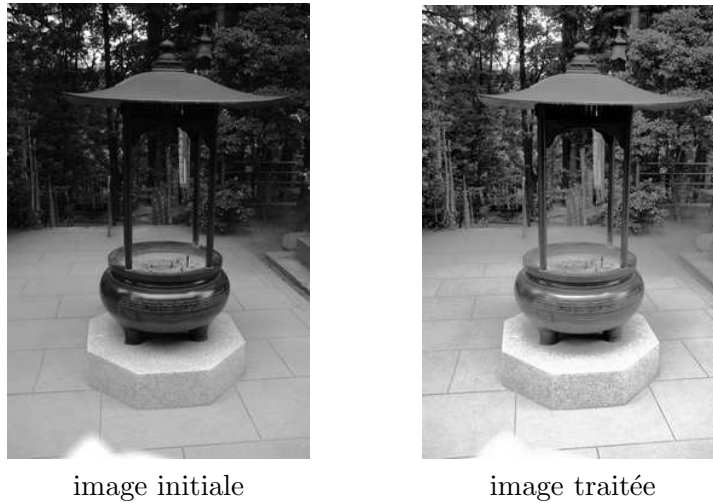


FIG. 2 – Exemple

2 Image

Une image est composée de pixels (picture element) répartis sous forme de tableau. On attribut à chaque pixel un niveau de gris codé en binaire sur n bits. S’il s’agit d’une image couleur, chaque pixel a trois composantes : une composante rouge, une composante verte et une composante bleue. A partir de ces trois composantes, il est possible de représenter plus ou moins bien la quasi-totalité du spectre visible.

Dans le cadre de ce projet, les images seront en couleurs et chaque niveau de gris sera codé sur un octet soit huit bits. Les valeurs de niveaux de gris varieront donc de 0 à $2^8 - 1 = 255$. Si l’image traitée est en “noir & blanc”, les composantes rouge, verte et bleue auront la même valeur pour un pixel donné. Le paragraphe 7.1 détaille le format de stockage des images utilisées pour ce projet.

3 Histogramme

3.1 principe général

Un histogramme est un ensemble de données statistiques permettant de représenter la distribution des intensités lumineuses des pixels d’une image, c’est-à-dire le nombre de pixels pour chaque niveau de gris. Pour une image en couleur, il est possible soit de faire un histogramme par composante (rouge, vert et bleue), soit de faire l’histogramme de la moyenne des trois composantes pour chaque pixel.

3.2 interprétation d'un histogramme

Un histogramme donne des informations sur la distribution des couleurs dans une image. Il permet par exemple de savoir si une image est plutôt sombre lorsque les pics se trouvent plutôt vers la gauche (figure 3). Pour une image plutôt claire, les pics de l'histogramme se trouveront vers la droite (figure 4).

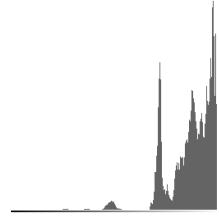
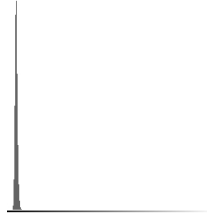


FIG. 3 – image sombre

FIG. 4 – image claire

Si la plupart des niveaux de gris sont situés vers le milieu de l'histogramme, cela signifie que l'image n'est pas très contrastée (figure 5). Enfin, si les niveaux de gris sont situés sur les extrêmes de l'histogramme, cela peut laisser penser que le contraste est trop fort (figure 6).

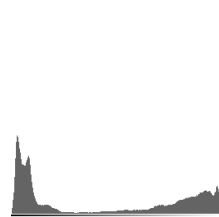
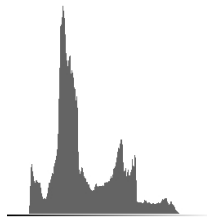


FIG. 5 – pas assez de contraste

FIG. 6 – contraste trop fort

3.3 construction d'un histogramme

Pour construire un histogramme, il faut d'abord créer un tableau H de taille k où k est le nombre de niveaux de gris différents. Les valeurs de H doivent être initialisées à 0. Il suffit alors de parcourir chaque pixel de l'image et d'ajouter 1 dans la case correspondant au niveau de gris du pixel traité.

4 Rééquilibrer un histogramme

Rééquilibrer un histogramme consiste à créer une image dont la luminance est répartie de façon homogène sur l'ensemble des niveaux de gris afin de mettre en valeur la dynamique des couleurs. Ce principe est illustré sur la figure 7 où $H(p)$ représente l'histogramme avant traitement et $G(p)$ après. Le contraste de l'image est alors augmenté pour les niveaux de gris avoisinant les maxima de l'histogramme. Inversement, le contraste est diminué pour les niveaux de gris avoisinant les minima de l'histogramme.

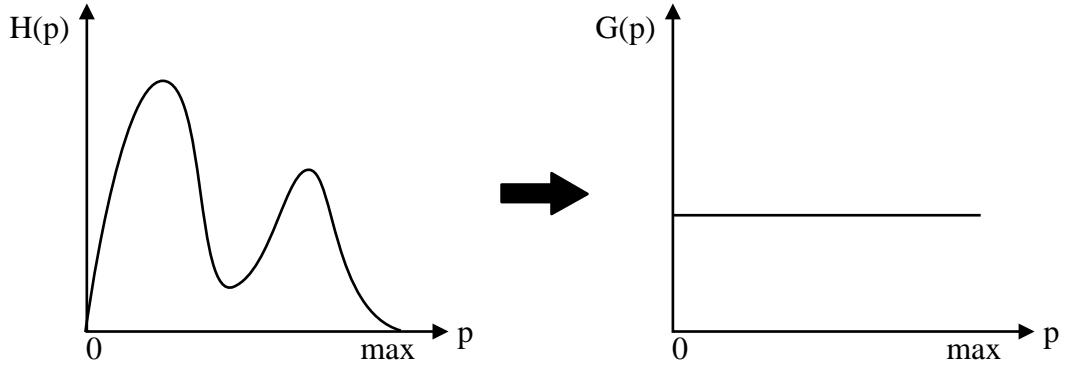


FIG. 7 – Rééquilibrer un histogramme

Pour rééquilibrer un histogramme, il faut donc trouver la transformation T sur les niveaux de gris des pixels de l'image d'origine telle que la répartition des pixels de l'image d'arrivée soit uniforme sur l'intervalle $[0, k - 1]$ où k représente le nombre de niveaux de gris codables sur n bits. L'histogramme d'une image peut être considéré comme fonction discrète de probabilité de densité. La transformation T doit donc vérifier :

$$\sum_{i=0}^{k-1} G(p_i) = \sum_{i=0}^{k-1} H(p_i) \quad (1)$$

Soit w la largeur de l'image traitée et h sa hauteur. Idéalement, l'histogramme rééquilibré $G(p)$ correspond à la fonction de probabilité de densité uniforme f telle que f soit constante :

$$G(p_i) = f = \frac{w \cdot h}{k - 1} \quad \forall i \in [0, k - 1] \quad (2)$$

Il s'agit d'une configuration idéale qui ne peut être obtenue que sur des histogrammes continus (et non discrets). En réinjectant l'équation (2) dans l'équation (1), on obtient alors :

$$\frac{w \cdot h}{k - 1} \int_0^p ds = \frac{w \cdot h}{k - 1} p = \int_0^p H(s) ds \quad (3)$$

où $p \in [0, k - 1]$. La transformation T peut alors s'écrire :

$$q = T(p) = \frac{k - 1}{w \cdot h} \int_0^p H(s) ds \quad (4)$$

L'équation (5) définit la transformation $T(p)$ sur l'intervalle $[0, k - 1]$.

Concrètement, les histogrammes utilisés sont discrets, il faut donc réécrire l'intégrale de l'équation (5) sous forme d'histogramme cumulé H_c :

$$\int_0^p H(s) ds \quad \Leftrightarrow \quad \sum_{i=0}^p H(i) = H_c(p) \quad (5)$$

Les figures 8 et 9 montrent une image et son histogramme avant et après qu'ils aient été rééquilibrés. Des détails invisibles sur l'image de départ sont dévoilés sur l'image traitée.

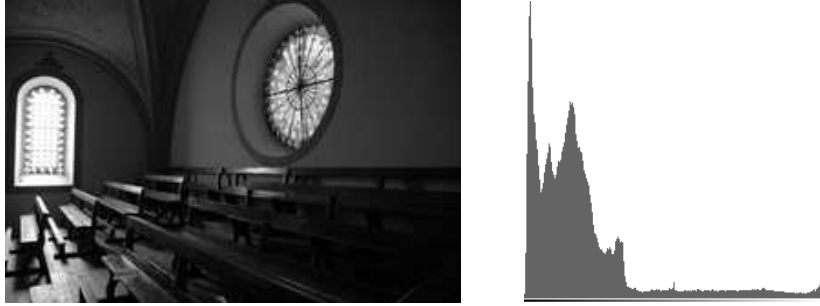


FIG. 8 – image originale et son histogramme

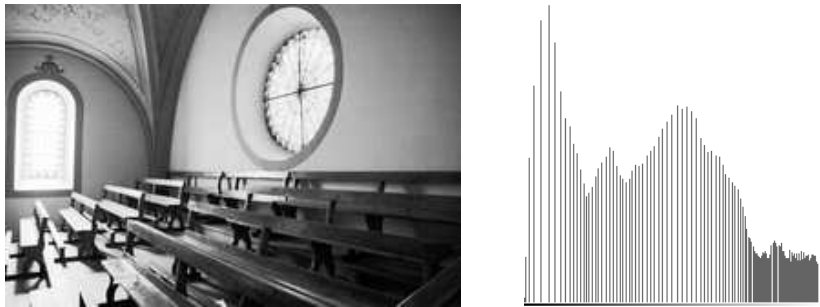


FIG. 9 – image rééquilibrée et son histogramme

5 Look-up Table

Une look-up table est un tableau permettant d'effectuer des transformations sur les couleurs d'une image. A chaque niveau de gris défini dans l'image de départ est associé un niveau de gris d'arrivée. Une look-up table n'est donc pas nécessairement bijective et deux niveaux de gris de départ peuvent se voir attribuer le même niveau de gris d'arrivée. D'ailleurs, le nombre de niveaux de gris défini dans l'image de départ n'est pas forcément égal à celui défini dans l'image d'arrivée. L'utilisation de look-up tables est courante dans les logiciels de traitement d'images car elle permet de combiner plusieurs opérations sur une image sans la détériorer au cours du traitement.

L'usage de look-up table est tout à fait adéquat pour l'emploi de la fonction T . Celle-ci sera donc représentée par une look-up table.

6 Travail demandé

6.1 code

Le but du projet est d'implémenter en langage **C** un programme permettant de rééquilibrer l'histogramme d'une image. Ce programme devra générer sous forme de fichier *ppm* (cf. paragraphe 7.1) l'image rééquilibrée ainsi que les histogrammes des deux images. Votre programme sera composé de fichiers `.c` et `.h`. Il se compilera avec un `makefile` et comportera une option `-h` pour l'aide. Toutes les options ainsi que le nom de l'image à traiter seront spécifiées sur la ligne de commande et nulle part ailleurs. Votre programme devra compiler et fonctionner sous Linux.

6.2 rapport

Vous devrez justifier le choix de vos méthodes dans un rapport d'environ 5-10 pages (pas plus). Ce rapport est fait pour montrer ce que vous avez compris de votre projet. Il doit montrer que vous savez prendre du recul par rapport à votre travail. Voici quelques règles à suivre pour mieux orienter son contenu :

- Ne perdez pas de temps à réexpliquer le sujet du projet, l'enseignant le connaît déjà, faites seulement un bref résumé de quelques lignes. De manière plus générale, ne détaillez pas des méthodes déjà expliquées dans l'énoncé à moins que vous les ayez modifiées.
- Un rapport sert surtout à montrer comment vous avez fait face aux problèmes (d'ordre algorithmique). Certains problèmes sont connus (on en parle dans l'énoncé), d'autres sont imprévus. Montrez que vous les avez remarqués et compris. Donner la liste des solutions à ce problème et indiquez votre choix. Justifiez votre choix (vous avez le droit de dire que c'est la méthode la plus facile à coder).
- Il ne doit figurer aucune ligne de code dans votre rapport. Un rapport n'est pas un listing de votre programme où vous détaillez chaque fonction. Vous devez par contre détailler vos structures de données et mettre du pseudocode pour expliquer vos choix algorithmiques.
Il est autorisé d'utiliser des "raccourcis" tels que "initialiser le tableau `tab` à 0" plutôt que de détailler la boucle faisant la même chose.
- n'hésitez pas à mettre des images dans votre rapport pour illustrer vos propos et vos résultats.
- Enfin, il est **très important** de faire la liste de ce que vous avez fait, de ce qui fonctionne correctement et la liste des dysfonctionnements. Précisez quand il s'agit d'options que vous avez rajoutées en plus de ce qui était demandé.

8 options

Il est conseillé de rajouter à votre programme quelques options. En voici quelques unes :

- Proposer d'autres transformations associées aux look-up tables comme le négatif d'une image par exemple.
- Proposer différentes façons de traiter les images en couleurs.
- Travailler sur des images 16 bits.
- ...

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.