

Images : travail direct sur les pixels.

Objectifs : Savoir modifier une image, ou obtenir une information sur une image, en travaillant directement sur ses pixels. Travail sur les matrices avec matlab. Traitement simple sur une image numérique.

Fonctions matlab étudiées : `imread()`, `imagesc()`, `image()`, `colormap('gray')`, `imwrite()`.

Fonction fournie : `view3D()`

Contrôle du TP : Rédiger un rapport avec les réponses aux questions, les scripts utilisés. Ce rapport vous servira pour le TP d'évaluation, pour lequel internet sera coupé, mais vous aurez droit à vos rapports (et à vos scripts et fonctions).

Image en noir et blanc, en niveaux de gris. Négatif d'une image

1. Récupérez une image en couleurs, et acquérez-la dans matlab. Ceci se fait grâce à la fonction `imread()`, dont vous pouvez voir le fonctionnement en tapant `help imread` sous matlab. Elle permet d'importer une image dans une matrice matlab, que nous appellerons `I`. Regardez la dimension de `I` en faisant `size(I)`. Si vous avez deux nombres seulement. Il doit y avoir trois chiffres : pouvez-vous dire ce qu'ils représentent ?
2. Affichez l'image en utilisant la fonction `image()`. Vous disposez d'une autre fonction, `imagesc()`, qui fonctionne de manière similaire, mais qui étale les valeurs des pixels entre 0 et 255. Pour comprendre la différence, créez une petite matrice 4×4 , que vous appellerez `A`, dont les valeurs seront soit des 0, soit des 1, soit des 2. C'est une image 4×4 en niveau de gris. Affichez-la à l'aide des fonctions `image()` puis `imagesc()`. Expliquez dans votre rapport la différence entre ces deux fonctions.
3. On travaille de nouveau sur l'image en couleurs. Affichez la avec cette fois la fonction `view3D()`, qui n'existe pas sous matlab, mais que je vous fournis sur e-campus. Faites tourner l'image pour l'observer sous toutes ses dimensions ! Vous pouvez aussi visualiser `A` avec cette fonction.
4. Elle possède donc trois canaux, un pour le rouge, un pour le vert, un pour le bleu. Chacune de ces couches de couleur est une matrice de la taille de l'image. On souhaite créer une image **en niveaux de gris**, c'est-à-dire qui ne possède qu'un canal donnant le niveau de gris de chaque pixel (c'est une matrice plate). Cette image sera en noir et blanc. Pour cela :
 - Créez une matrice `I2` qui vaut 0.30 fois la matrice de rouge plus 0.59 fois la matrice de vert plus 0.11 fois la matrice de bleu.
 - (**Image en niveaux de gris.**) Visualisez la avec `imagesc`. Elle apparaît en rouge et bleu, car dans matlab, la palette de couleurs par défaut ne fait pas correspondre 0 à noir et 255 à blanc, mais 0 à bleu et 255 à rouge (ceci pour que les graphes 3D aient bel aspect). Pour changer cela, taper `colormap('gray')`. L'image doit alors apparaître en noir et blanc. Faites `size(I2)`. C'est bien une matrice plate.
5. (**Image en noir et blanc.**) On souhaite faire la même chose, mais *sans changer la structure de l'image* : on va créer une matrice `I3` qui contient l'image en noir et blanc, mais qui est toujours composée de 3 couches. Pour cela, il suffit de mettre, pour chaque pixel, la même valeur pour chacune des trois couches. Créez donc une matrice `I3`, identique à la matrice `I` contenant l'image en couleur, mais où à la place de la couche rouge, la couche verte, et la couche bleue, vous mettez une couche contenant la moyenne de ces trois couches.
6. Générez maintenant une image en couleur qui soit le négatif de votre image originale : chaque pixel est remplacé par un pixel de couleur complémentaire.
7. Faites une fonction qui prend en entrée deux images de même taille (en niveaux de gris) et qui ressort une image qui en est la moyenne : chaque pixel de l'image résultat est la moyenne des pixels correspondants des images de départ. Vous pouvez tester sur des visages par exemple (faites en sorte que les yeux soient placés au même endroit de l'image à peu près..).

Recherche sur les pixels de l'image. Création d'une image.

1. Écrire une fonction qui prend en entrée une matrice d'image en niveau de gris et retourne le nombre de pixels blancs (valeur 255).
2. Créez une fonction qui retourne la valeur maximale de niveau de gris et la valeur minimale de niveau de gris présentes dans une image. Si celle-ci est en couleurs, votre fonction commencera par la convertir en niveaux de gris.
3. Écrire une fonction qui prend en paramètre `n` et `m` ($n > m$), et permet de construire une image de taille $n \times n$, qui est blanche autour d'un carré noir de taille $m \times m$. Vous pourrez également faire en sorte que votre fonction sauvegarde l'image obtenue dans un fichier `carre.bmp` (il faudra utiliser `imwrite()`).