Traitement d'images- Matlab

Organisation

Les TPs s'effectuent en **monôme** chacun sur son ordinateur portable, avec le logiciel **Matlab** installé avant la première séance.

Matériel utilisé

Ordinateur portable avec le logiciel Matlab installé avant la première séance.

Absences

La présence en TP est **obligatoire** sauf absence justifiée.

Notations

La dernière séance (3h) sera réalisée sous forme d'un TP noté.

Comportement en TP

Vous êtes en TP pour appliquer vos connaissances. Avant d'appeler l'intervenant, essayer par vous-même, cherchez l'erreur, etc. Si cela ne fonctionne pas, ensuite appelez l'intervenant.

Recommandations

Décrire ce que vous faites. Analyser les résultats obtenus.

Exercice 0 : Rappels – Fonctions utiles

Dans ce TP, vous mettrez en œuvre les différents notions et algorithmes que vous avez rencontrés dans le cours afin d'interpréter des images.

1. Lecture d'une image

Pour pouvoir effectuer des traitements sur une image, il est nécessaire de charger celle-ci. Pour cela, nous pouvons utiliser les fonctions Matlab suivantes :

$imread$: pour lire une image

$imshow$: pour afficher une image

$title$: pour donner un titre à l'image affichée

2. Taille de l'image

Pour récupérer la taille de l'image, nous pouvons utiliser la fonction Matlab suivante :

$size$: récupérer la taille d'une image

3. Écriture d'une image

Afin d'enregistrer une image sur le disque, nous pouvons utiliser la fonction Matlab suivante :

$imwrite$

4. Conversion d'une image couleur en niveaux de gris

Pour convertir une image couleur en une image niveaux de gris, nous pouvons utiliser la fonction Matlab suivante :

$$image\_gris = rgb2gray(image\_couleur);$$

5. Accès aux pixels d'une image

Pour récupérer la valeur d'un pixel gris, il faut faire :

$$Pixel = image(i,j)\ ;\ avec\ i = ligne\ et\ j = colonne$$

6. Histogramme d'une image

Pour représenter la distribution des valeurs de niveaux de gris d'une image, nous pouvons afficher son histogramme en utilisant la fonction Matlab suivante :

$imhist(image\_gris)$;

7. Image binaire

En traitement d'images, le seuillage est une technique simple appliquée pour réduire la dynamique d'une image en niveaux de gris. Le seuillage le plus utilisé est le seuillage binaire. En effet, ce type de seuillage permet de ramener la dynamique d'une image à deux niveaux. Pour faire cela, plusieurs méthodes peuvent être utilisées :

- Utiliser des boucles et/ou des instructions : $for - if - elseif - else - etc.$

$$nouvelle_{valeur_{pixel}} = \begin{cases} K_1, & si\ ancienne_{valeur_{pixel}} \leq Seuil \\ K_2, & si\ ancienne_{valeur_{pixel}} > Seuil \end{cases}$$

- Utiliser les fonctions Matlab suivantes : $im2bw$ ou $imbinarize$

8. Étiquetage

L'analyse des composants connectés (étiquetage) est une opération qui permet de grouper tous les pixels connectés dans une image binaire sous une même étiquette. Pour faire cela, plusieurs méthodes peuvent être utilisées :

- Méthode détaillée dans le cours
- Utiliser la fonction Matlab suivante : $bwlabel$

Exercice 1 : Voisinage

Le but de cet exercice est de construire une animation du jeu de la vie. En effet, on part avec un espace $2D$ composé de $H \times W$ cellules (image $2D$ de taille $H \times W$) qui sont aléatoirement vivantes (pixels allumés = 1) ou mortes (pixels éteints = 0). A chaque génération (itération), cette population évolue en suivant les deux règles suivantes :

- Une cellule morte devient vivante si elle est entourée exactement par 3 cellules vivantes parmi les 8 cellules qui l'entourent.
- Une cellule vivante survit seulement si elle est entourée par deux ou trois cellules vivantes.

Visualiser les différentes étapes.

Exercice 2 : Contours

La détection de contours dans une image numérique est l'une des plus importantes opérations utilisées en traitement d'images. En effet, les méthodes de détection de contours sont nombreuses. La plupart de ces méthodes sont basées sur les dérivées premières et secondaires de l'image. Le but de cet exercice est d'écrire un algorithme simple qui consiste à détecter les contours sur une image donnée.

- Lire l'image « **cameraman.jpg** ». Quel est le type de cette image ?
- Convertir cette image en une image niveaux de gris.
- Détecter les pixels contours en utilisant le filtre de « **Canny** ».
- Réduire le nombre de pixels faux contours (fausses alarmes) en utilisant la méthode basique de seuillage.
- Colorier en rouge les pixels contours conservés sur l'image originale.
- Faire la même chose en utilisant le filtre : « **Sobel** », « **Prewitt** ». Commentaires.

Afficher tous les résultats.

Pour la détection de contours, vous pouvez utiliser la fonction Matlab suivante : edge

Exercice 3 : Détection de couleur

L'objectif de cet exercice est de savoir comment détecter une couleur précise dans une image RGB (par exemple la couleur jaune).

- Lire l'image « **fleurs.jpg** ». Quel est le type de cette image ?
- Extraire les 3 composantes « **R** », « **G** » et « **B** ». Afficher chacune de ces 3 composantes.
- Afficher l'histogramme des 3 composantes « **R** », « **G** » et « **B** ».
- Afficher les 3 images binaires (**BW_R**, **BW_G** et **BW_B**) obtenues en appliquant la méthode de seuillage binaire sur chacune des 3 composantes. ***Attention aux choix des valeurs des seuils. On cherche à détecter la couleur jaune.***
- Combiner ou multiplier les 3 images binaires des 3 composantes « **R** », « **G** » et « **B** ». Afficher le résultat obtenu « **Mask** ». *Attention ! Ce dernier est une image binaire qui contient uniquement les pixels jaunes.*
- Appliquer le Mask obtenu à l'image originale. Afficher le résultat final.

Exercice 4 : Étiquetage

Dans cet exercice, on vous donne l'image couleur « **chiffres.png** ». Il s'agit d'écrire un programme qui permet de :

- Convertir cette image couleur en une image niveaux de gris.

- Faire une binarisation de l'image niveaux de gris en appliquant la méthode de seuillage binaire.
- Étiqueter l'image binaire obtenue. Préciser le nombre d'objets contenus dans cette image.
- Si vous avez le temps, colorier tous les objets étiquetés dans l'image (une couleur par objet).
- Afficher tous les résultats obtenus.

Exercice 5 : Morphologie mathématique

Le but de cet exercice est de vous montrer les effets des quatre opérateurs de la morphologie mathématique (dilatation, érosion, ouverture, fermeture). L'image qu'on cherche à traiter dans cet exercice est « **morp.tif** ». En effet, il s'agit d'écrire un programme permettant d'extraire uniquement l'objet le plus clair de l'image. Pour faire cela, plusieurs étapes sont à suivre :

- Afficher l'image « **morp.tif** ». Quel est le type de cette image ?
- Afficher l'histogramme de cette image.
- Binariser cette image de façon à garder les pixels les plus clairs : « **BW** »
- Appliquer les opérateurs de la morphologie mathématique :
  - Érosion de « **BW** » en utilisant un élément structurant carré de taille 2 : « **BW_erodee** ».
  - Dilatation « **BW** » en utilisant un élément structurant carré de taille 2 : « **BW_dilatee** ».
  - Ouverture « **BW** » en utilisant un élément structurant carrée de taille 2 : « **BW_ouver** ».
  - Fermeture de « **BW_ouver** » en utilisant un élément structurant carré de taille 10 : « **BW_ferm** ».

Le résultat obtenu « **BW_ferm** » est une image binaire qui contient uniquement les pixels de l'objet le plus clair de l'image.

Multiplier l'image originale avec l'image binaire. Commentaires.

Vous pouvez utiliser les fonctions Matlab suivantes :

$strel$: pour créer un élément structurant

$imerode$: pour effectuer une érosion

$imdilate$: pour effectuer une dilatation

$imopen$: pour effectuer une ouverture

$imclose$: pour effectuer une fermeture

Bon travail