TD 10 : CLASSIFICATION

EXERCICE 1 : Corrélation

Dans cet exercice, on programme la corrélation entre une imagette représentant une forme de référence et une image à analyser.

➢ Etudier le script `TD10Ex1.m`, et compléter les parties manquantes. Vous utiliserez la fonction `corr2` (voir le help de Matlab).

```
%% Effectuer la corrélation sur toute l'image testée
% et créer une image de corrélation
rescorr = zeros(H,W);
for i=1:H-H1
   for j=1:W-W1
      c = corr2(ref,-- A REMPLIR));
      rescorr(-- A REMPLIR,-- A REMPLIR)=c;
   end;
end;
```

↳ *Expliquer l'algorithme. Que représente l'image rescorr ? Quelles sont ses valeurs min et max ?*

↳ *Comment déduire de cette image la présence du chiffre 8 ? Quelles sont les coordonnées des chiffres 8 détectés ?*

↳ *Multiplier l'image de référence par un coefficient, par exemple 0.7. Commenter*

↳ *Relancer le programme avec l'image de référence 8n.tif. Commenter.*

↳ *Relancer le programme avec l'image de référence 8b.tif. Commenter.*

EXERCICE 2 : Analyse d'image par « template matching »

Nous allons maintenant effectuer la lecture automatique d'une image contenant des colonnes de chiffres. Pour cela, nous devons développer un algorithme qui réalise la labellisation des chiffres, la reconnaissance de chaque chiffre détecté, l'analyse de leurs positions relatives pour reconstituer les nombres.

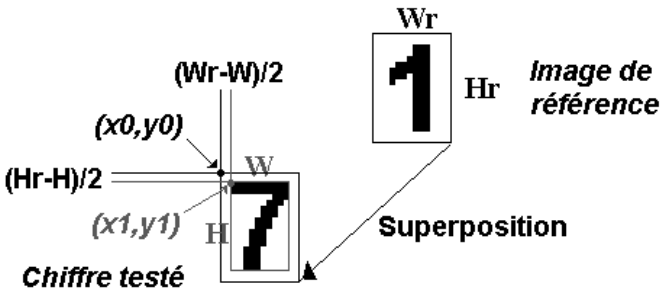
➢ Ouvrir le fichier `td10ex2.m`. Ce fichier :

- Ouvre l'image à analyser *'chiffre.tif'*

- Charge toutes les images de référence dans le tableau iref : iref(:, :,k) est l'image de référence du chiffre k.

- Initialise une structure stockant les résultats de corrélation :

```
res = zeros(nbobjets,4);
res(i,1) :% indice de l'image de reference détectée
res(i,2) :% coordonnée x (ligne) du point central
res(i,3) :% coordonnée y (colonne) du point central
res(i,4) :% score corrélation
```



Conventions de notation

➢ Réaliser la reconnaissance sur chaque objet labellisé en complétant la boucle for sur tous les objets détectés :

```
for i=1:nbobjets
    …
end;
```

↳ *Compléter l'algorithme en vous basant sur la figure ci-dessus. La reconnaissance est-elle exacte ? Quels sont les scores de corrélation obtenus ?*

↳ *Tester le programme sur l'image chiffres3.tif. Fonctionne-t-il toujours ? Comment expliquez-vous les confusions ?*

EXERCICE 3 : Reconnaissance de chiffres manuscrits – Etape (4) - Approche du plus proche voisin

Dans l'exercice 1, nous avons réalisé la localisation des chiffres et l'extraction de paramètres caractéristiques, les cavités. Ces tâches sont désormais assurées par les fonctions `localise.m` et `cavites.m`.

Nous allons maintenant réaliser la classification par la méthode des plus proches voisins. Il nous faut donc dans un premier temps constituer les bases d'apprentissage et de test. Les images 3e1, 4e1 et 5e1 sont utilisées en apprentissage, les images 3e2, 4e2 et 5e2 en test.

➢ Lancer le programme `traite.m` puis `base.m`. On crée ainsi le fichier ***bases.mat*** qui sauvegarde les matrices suivantes :

LearnIn(:,i) : vecteur (5x1) de paramètres (les 5 cavités) du chiffre numéro i

LearnClass(i) : classe (scalaire : 3, 4, ou 5) du chiffre numéro i.

LearnPos(:,i) : vecteur (4x1) des positions des coins du rectangle englobant le chiffre i
Et l'équivalent pour la base de test avec les matrices TestIn, TestClass et TestPos

- Taper :

```
clear all
load bases
who
```

- *Vérifiez que vous obtenez bien les matrices ou vecteurs mentionnés. Combien y a-t-il d'exemples d'apprentissage et d'exemples de test ?*

- Ouvrir maintenant le fichier `TD10Ex3.m` réalisant la classification des chiffres de la base de test.

- *Programmer la classification par la méthode des Plus Proches Voisins en complétant le cœur du programme.*

- *Quel est le taux de reconnaissance ? Que représente la matrice conf ?*

- *Commentez les résultats en vous aidant des images de résultats (le nombre de petits carrés incrustés à gauche des chiffres mal classés indique la classification obtenue pour ces chiffres), et des images de cavités (images cav3e1.tif, cav3e2.tif, cav4e1.tif, etc.)*

- *Comment pourrait-on améliorer ces résultats ?*

EXERCICE 4 : Reconnaissance de chiffres manuscrits – Etape (4) – Approche Bayésienne

Nous reprenons les données de l'exercice précédent, mais nous allons maintenant proposer une approche bayésienne. Nous supposons que les fonctions de densité de probabilité des vecteurs d'attributs conditionnellement aux classes sont gaussiennes. L'apprentissage va consister à apprendre les paramètres de ces distributions sur la base d'apprentissage.

- Lancer le programme `traite.m` puis `base.m`. On crée ainsi le fichier ***bases.mat*** qui sauvegarde les matrices suivantes :
  LearnIn(:,i) : vecteur (5x1) de paramètres (les 5 cavités) du chiffre numéro i
  LearnClass(i) : classe (scalaire : 3, 4, ou 5) du chiffre numéro i.
  LearnPos(:,i) : vecteur (4x1) des positions des coins du rectangle englobant le chiffre i
  Et l'équivalent pour la base de test avec les matrices TestIn, TestClass et TestPos

- *Combien de fonctions de densités de probabilité faut-il apprendre ? Quels sont les paramètres à apprendre et leurs dimensions ?*

- Ouvrir maintenant le fichier `TD10Ex4.m`. La première partie du fichier permet de réaliser l'apprentissage.

↳ *Compléter les parties manquantes : calcul de Vm, C, Cinv, D.*

➢ La seconde partie concerne la reconnaissance des chiffres de la base de test.

↳ *Programmer la règle de décision*

↳ *Quel est le taux de reconnaissance ?*

➢ Le calcul de l'inverse d'une matrice est souvent délicat. Plutôt que de calculer l'inverse de la matrice de covariance, il est souvent préférable d'utiliser l'opérateur de division matricielle.

↳ *Taper dans la fenêtre de commande* `help slash` *et modifier le calcul des densités de probabilité dans votre fichier.*

↳ *Quel est le nouveau taux de reconnaissance ?*

↳ *Quelles sont les performances obtenues par rapport à la règle du ppv ?*

EXERCICE 5 : Reconnaissance de symboles musicaux, approche Bayésienne

On souhaite classer les symboles musicaux segmentés dans l'image d'une partition musicale, afin de faire la reconnaissance automatique de la partition (OMR, Optical Music Recognition). La base de données se trouve dans le répertoire OMR, avec 3 sous-répertoires contenant respectivement les images de bémols, bécarres et dièses. Chaque sous-répertoire contient `nbImages = 100` images. On utilisera les *`nbLearn`* premières pour l'apprentissage automatique.

Ouvrir le fichier TD10Ex5.m.

La première étape consiste à extraire un vecteur de caractéristiques. On propose d'utiliser les moments centrés normés.

↳ *Justifiez ce choix*

↳ *Soit l'image I de dimensions* $H \times W$. *Montrez que l'on peut calculer matriciellement le moment d'ordre p,q* par

$$m_{pq} = [1^p \; 2^p \quad \cdots \quad H^p] I \begin{bmatrix} 1^p \\ 2^p \\ 1^p \\ W^q \end{bmatrix}$$

➢ Calculez les moments centrés normés jusqu'à l'ordre 3 et stockez les dans la base de données. On définira les structures de données suivantes :

- `BD{k}` tableau `nbFeatures` × `nbImages` pour la classe *k*
- `BD{k}(:,1:nbLearn)` tableau pour les exemples d'apprentissage de la classe k

- `BD{k}(:,nbLearn+1:nbImages)` tableau pour les exemples de test de la classe *k*

↳ *Observer la répartition des attributs ? Vous paraissent-ils pertinents ?*

➢ Complétez l'apprentissage bayésien

➢ Complétez la phase de test

↳ *Quels sont les résultats de généralisation ? Sont-ils satisfaisants ? Observez les images mal classées. Est-il possible d'améliorer les résultats (optimiser en diminuant le nombre de caractéristiques, augmenter le nombre de caractéristiques, etc.) ?*