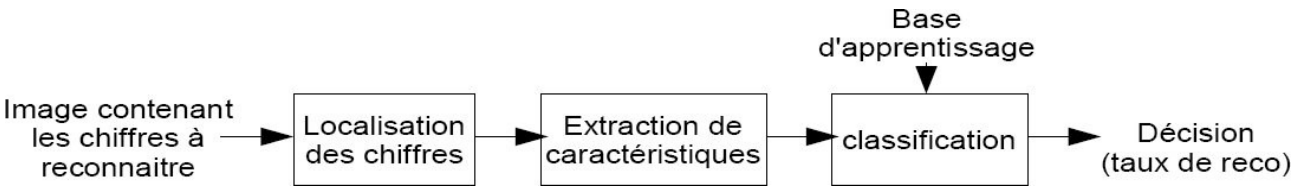
Reconnaissance de Formes

Clément Chatelain

Objectifs:

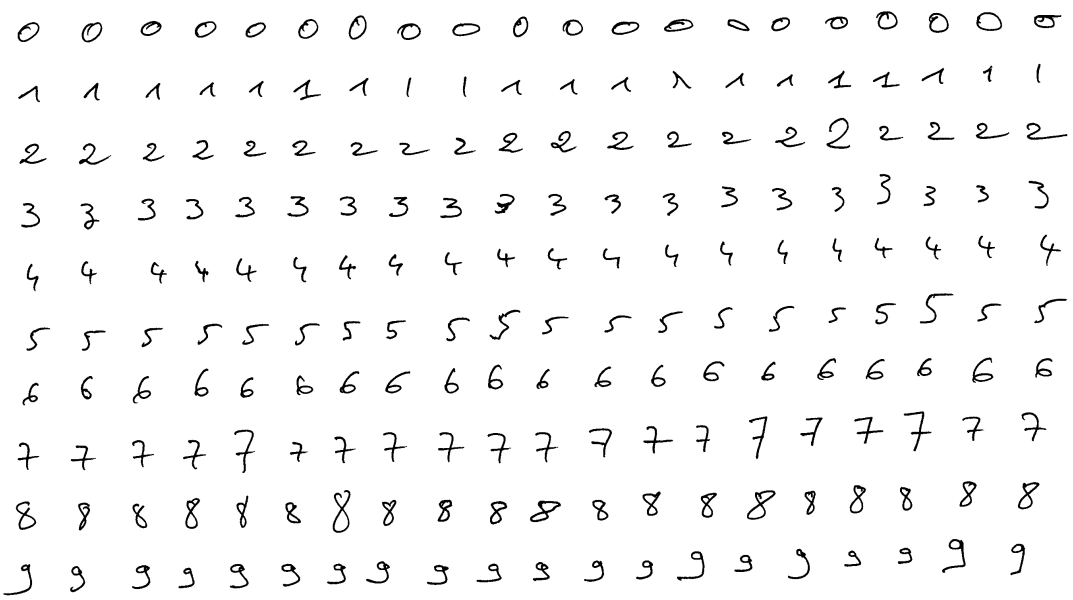
Appréhender les différentes étapes d'un système de RdF à travers le développement d'une chaîne de reconnaissance de chiffres manuscrits sous MATLAB. Le schéma du système complet sera le suivant:



Pour vous aider, un squelette de programme vous est donné (*squelette.m*)

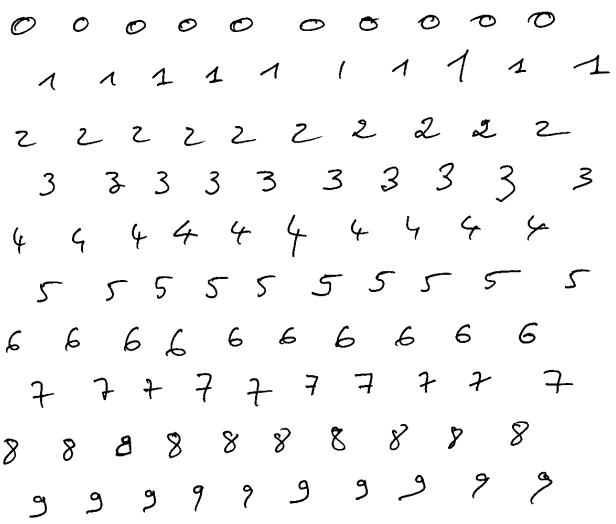
1- **Les données (disponibles sur** moodle**)**

On dispose dans le fichier image "app.bmp" d'une image binaire contenant 200 chiffres manuscrits (20 exemples par classe):



Les chiffres contenus dans cette image une fois extraits serviront de base d'apprentissage pour les deux classifieurs à développer.

Le fichier image "test.bmp" est une image binaire contenant 100 chiffres (10 par classe) et permettant de tester le système de reconnaissance de formes développé.



2- Prétraitements: localisation/extraction des chiffres manuscrits

Cette partie vous est donnée : les images d'apprentissage et de test sont parcourues afin de localiser et d'extraire les imagettes de chiffres. Ces traitements sont effectués par les fonctions *subimage.m*, *crop.m*, *extractionLignes.m* et *extractionImages.m*, qui sont appelées par *squelette.m*.

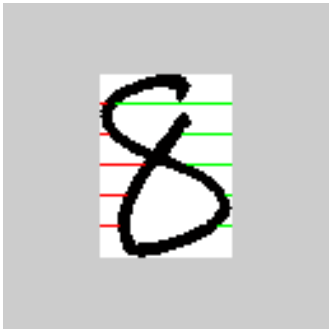
3- Extraction de caractéristiques

Vous êtes totalement libres de développer les caractéristiques qui vous sembleront pertinentes pour reconnaître les chiffres. Voici toutefois deux idées de vecteurs simples à utiliser :

Profils :

Pour cette première méthode de reconnaissance, les caractéristiques à extraire des chiffres sont leurs profils gauche et droit. Ces mesures sont obtenues de la façon suivante: on détermine sur un certain nombre de lignes –réparties uniformément sur la hauteur du chiffre – la distance entre le bord gauche (resp. droit) du chiffre et le premier pixel noir rencontré sur cette ligne; l'ensemble de ces distances définit un profil gauche (resp. droit) du chiffre. On peut alors constituer un vecteur à *d* composantes en concaténant les deux profils (*d/2* distances par profil).

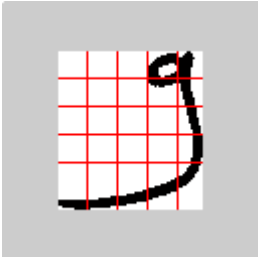
Illustration : les caractéristiques sont les longueurs des traits rouges et verts.



Conseil de développement : écrire une fonction *extraitProfils* prenant en entrée une imagette dont on souhaite extraire les profils gauche et droit et la valeur de *d*. Cette fonction doit renvoyer un vecteur à *d* composantes. On pourra choisir dans un premier temps *d/2*=5 (cf. exemple). Les profils gauche et droit devront être normalisés (en les divisant par exemple par la largeur de l'image du chiffre traité) puisque les chiffres ne sont pas tous de même taille.

Zoning et densités de pixels :

Les caractéristiques utilisées dans cette deuxième méthode de reconnaissance sont basées sur la densité de pixels noirs calculées dans différentes zones (zoning) de l'image du chiffre. Ces mesures sont obtenues de la façon suivante: on "découpe" horizontalement et verticalement le rectangle englobant du chiffre en zones de taille égale; le nombre de pixels noirs dans chaque zone forme alors les composantes du vecteur de caractéristiques. En découpant par exemple l'image en *n* zones verticales (d'égale largeur) et *m* zones horizontales (d'égale hauteur), on obtient un vecteur à *nxm* composantes.



0	0	0	133	120
0	0	0	15	80
0	0	0	0	76
0	0	0	0	91
65	66	78	81	28

Conseil de développement : écrire une fonction *extraitDensites* qui prend en entrée l'image d'apprentissage dont on souhaite extraire les densités. On pourra choisir dans un premier temps $n=m=\sqrt{d}=5$. Les densités dans chaque zone devront être normalisées (en les divisant par exemple par la surface de la zone) puisque les chiffres ne sont pas tous de même taille.

4- Classification

Ici aussi, vous êtes libres de développer la ou les méthode(s) de classification qui vous semblera la plus adaptée. Toutefois, vous devrez au minimum développer un des deux classifieurs décrits ci-après. Idéalement, une combinaison de ces deux classifieurs sera réalisée.

a) classifieur par distance euclidienne minimum

La méthode de décision proposée consiste à attribuer au point de R^d, représentant la forme à reconnaître, la classe dont la distance du point au centre de cette classe est minimale. Il faut donc disposer, préalablement à toute reconnaissance, du centre de chacune des classes: c'est l'objet de la phase d'apprentissage. On pourra transformer les distances entre points en vecteurs de probabilité a posteriori par l'estimation suivante:

$$\hat{p}(\omega_i / x) = \frac{\exp(-d(x, \omega_i))}{\sum_{j=0}^{9} \exp(-d(x, \omega_j))} .$$

b) kppv

La méthode de décision proposée (K plus proches voisins) nécessite de disposer, préalablement à toute reconnaissance, de l'ensemble des points formant la base d'apprentissage. La règle des Kppv est alors la suivante: étant donnée une distance sur R^d, calculer la distance du point à classer, représentant la forme à reconnaître, à tous les points de la base d'apprentissage et affecter à ce point la classe la plus représentée parmi ses K plus proches voisins.

5- Analyse et conclusion

Evaluer les performances de votre système (les taux de reconnaissance en Top1, Top2 et Top3, matrice de confusion, etc.) en fonction des différents classifieurs et jeux de caracs utilisés.

Quels sont les meilleurs classifieurs ? Les meilleurs caractéristiques ? Comment faire mieux ?