Rapport de TP
FI3-INFO501 Année 2021-2022
Polytech Annecy-Chambéry – Université Savoie Mont-Blanc

DORNIER Lisa
LEGLISE Cloé
24/11/2021

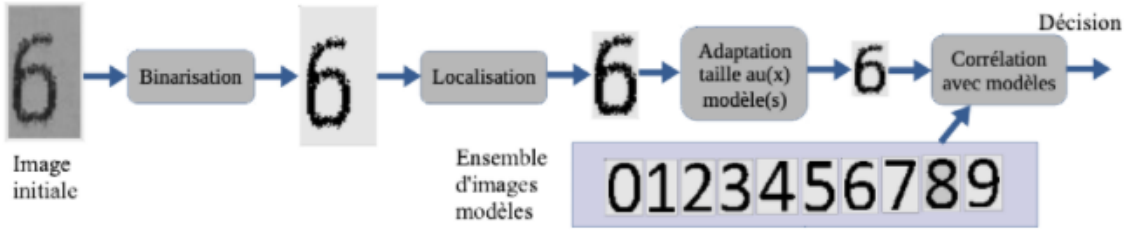
Rapport du TP2 – Lecture automatique de chiffres par analyse d'image

I. Introduction

Les objectifs de ce TP sont les suivants :

- Savoir illustrer la reconnaissance par corrélation avec des modèles
- Comprendre la reconnaissance automatique des caractères

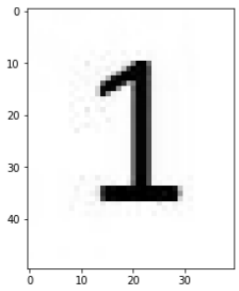
Les différentes étapes de ce TP sont représentées par le schéma suivant :



Nous commençons par récupérer le code principal dans le dossier src/ et nous ajoutons à tous cela le dossier tests qui contient les tests unitaires ainsi que le dossier assets qui contient des images de modèles et de tests.

II. Travail préparatoire

Nous commençons par exécuter le fichier main.py afin de vérifier qu'une image s'affiche bien à l'écran :



Question 1 :

Nous analysons ensuite la classe image, nous retrouvons 4 méthodes :

- Une initialisation d'une image composée d'un tableau de pixel et de deux dimensions (H et W).
- Le remplissage du tableau avec l'affectation des dimensions de l'image
- La lecture d'une image à partir d'un fichier existant
- L'affichage à l'écran d'une image

Après avoir compris cette classe, nous l'avons ensuite copiée dans notre code.

Question 2 :

Nous écrivons maintenant une méthode binarisation(self, S) dans la classe image afin de passer d'une image codée sur 256 valeurs à une image avec seulement deux valeurs (0 ou 255). Pour cela, nous allons comparer pour chaque pixel de l'image la valeur du pixel à une valeur choisie par l'utilisateur (S). Il faut itérer sur tous les pixels du tableau numpy et comparer la valeur au seuil. Le résultat est donné sous forme d'une nouvelle image que l'on crée dans la fonction afin de ne pas modifier l'image de base.
Nous avons le code suivant :

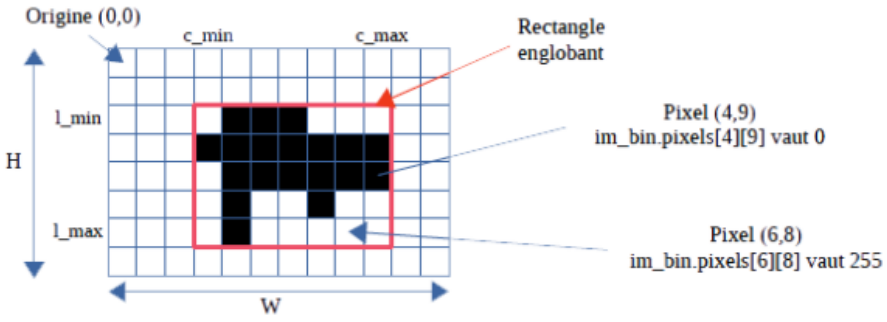
```
51
52      def binarisation(self, S):
53          #création d'une image aux dimensions voulues :
54          im_bin = Image()
55          im_bin.set_pixels(np.zeros((self.H, self.W), dtype = np.uint8))
56          #parcours de chaque pixel par ligne et par colonne :
57          for i in range(self.H):
58              for j in range(self.W):
59                  #choix de l'action à effectuer pour chaque pixel :
60                  if self.pixels[i][j] >= S:
61                      im_bin.pixels[i][j] = 255
62                  else :
63                      im_bin.pixels[i][j] = 0
64          #renvoi de l'image binarisée :
65          return (im_bin)
66
```

Explication :

Nous attribuons un tableau de 0 de la bonne dimension à im_bin. Ensuite nous parcourons les colonnes de im_bin et les lignes de im_bin afin de tester chaque pixel. Nous cherchons à savoir si nous voulons garder le pixel en noir ou si nous voulons l'ignorer et le passer en blanc. Si le pixel est en dessous du seuil, nous le passons en blanc (0), sinon nous le passons en noir (255).

Question 3 :

Nous allons écrire maintenant une méthode localisation(self) dans la classe image. Cette méthode va calculer et retourner l'image recadrée sur le chiffre à identifier. Sur une image de dimension H × W, nous allons déterminer les coordonnées l_{min}, l_{max}, c_{min} et c_{max} du rectangle englobant la forme noire (valeurs 0). Nous construirons ensuite une image limitée à ce rectangle.
Le schéma ci-dessous explique la méthode :



Nous avons le code suivant :

```
    def localisation(self):
        #initialisation des curseurs :
        c_max = 0
        c_min = self.W
        l_max = 0
        l_min = self.H
        #parcours de chaque pixel de l'image  :
        for l in range(self.H):
            for c in range(self.W):
                #encadrement par les curseurs :
                if self.pixels[l][c] == 0 :
                    if c < c_min :
                        c_min = c
                    elif c > c_max :
                        c_max = c
                    if l < l_min :
                        l_min = l
                    elif l > l_max :
                        l_max = l
        nb_c = c_max - c_min
        nb_l = l_max - l_min
        #création d'une nouvelle image aux dimensions voulues :
        image = Image()
        image.set_pixels(np.zeros((nb_l, nb_c), dtype = np.uint8))
        #remplacement des pixels de la nouvelle image par les pixels voulus :
        for l in range(nb_l):
            for c in range(nb_c):
                image.pixels[l][c] = self.pixels[l_min+l][c_min+c]
        #renvoi de l'image correctement recadrée :
        return (image)
```

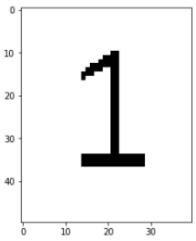
Explication :

Dans cette méthode, nous commençons par initialiser les curseurs l_{min}, l_{max}, c_{min} et c_{max} afin de parcourir toute l'image et de tester la couleur de chaque pixel. Une fois que nous avons parcouru toute l'image, nous connaissons ses délimitations. A partir de ce moment, nous créons une nouvelle image avec les dimensions trouvées et nous remplaçons chaque pixel de cette image par le pixel correspondant à la bonne partie de l'image binarisée.

III. Reconnaissance automatique de chiffre :

Question 1 :

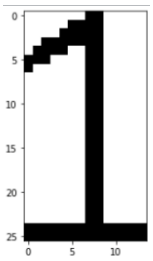
On lance le fichier main.py afin de tester notre méthode binarisation. Nous testons avec différentes valeurs de seuil et nous observons le résultat suivant :



Nous lançons également le fichier de tests test_Image.py afin de voir si notre méthode répond bien aux spécifications demandées. Cela ne nous renvoie pas d'erreur.

Question 2 :

Nous faisons la même chose que dans la question 1 afin de tester notre méthode localisation. Nous observons le résultat suivant :



Nous lançons également le fichier de tests test_Image.py. Nous n'obtenons pas d'erreurs.

Question 3 :

Durant cette question, nous utiliserons la fonction resize de la librairie skimage, cela nous permettra de donner à une image des dimensions imposées. Nous l'utiliserons de la manière suivante :

```
resize(im.pixels, (new_H,new_W), 0)
```

Avec :

- "im": L'image dont on veut modifier les dimensions
- "new_H" et "new_W" : Nouvelles dimensions de l'image
- "0" : Indique l'absence d'interpolation (permet de conserver une image binaire si l'image de départ est binaire)
- "im_resized" : L'image redimensionnée

On ajoute à la classe Image la méthode resize(self,new_H,new_W) qui redimensionnera l'image à la taille voulue et renverra un autre objet de type Image en sortie.

Nous avons le code suivant :

```
120     def resize(self, new_H, new_W):
121         #création d'une image :
122         image = Image()
123         #variable temporaire avec l'image aux dimensions souhaitées :
124         temporaire = resize(self.pixels, (new_H, new_W), 0)
125         #conversion de la variable temporaire en tableau d'entiers :
126         image.set_pixels(np.uint8(temporaire*255))
127         #renvoi de l'image :
128         return image
129
```

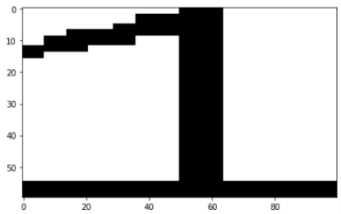
Explication :

Nous commençons par créer une image vide. On crée aussi une variable temporaire en utilisant la fonction resize de la librairie skimage. Nous obtenons un tableau de flottants et non plus un tableau d'entiers. Nous devons donc passer notre variable temporaire en un tableau d'entiers, pour cela, nous utilisons la commande suivante :

```
np.uint8(pixels_resized*255)
```

Nous injectons dans image notre variable temporaire convertie grâce à cette commande. Nous n'avons plus qu'à retourner l'image.

Nous testons dans le fichier main.py la fonction resize avec l'image obtenue par la localisation (on choisit des dimensions quelconques) :



Nous n'avons pas d'erreurs affichées lors des tests.

Question 4 :

Nous allons maintenant écrire la méthode similitude(self, image) qui mesurera la similitude par corrélation d'images entre les images self et image.
Nous obtenons le code suivant :

```
140     def similitude(self, im):
141         #initialisation du compteur de nombre de pixels identiques :
142         compteur = 0
143         #parcours de l'image :
144         for i in range (self.H):
145             for j in range(self.W):
146                 #comparaison de chaque pixel et modification du compteur :
147                 if self.pixels[i][j] == im.pixels[i][j]:
148                     compteur += 1
149         #calcul et renvoi de la similitude :
150         similitude = compteur/(self.H*self.W)
151         return similitude
152
```

Explication :

On commence par initialiser un compteur à 0. Ensuite, nous parcourons toute l'image et pour chaque pixel, nous regardons si le pixel de notre image a la même valeur que le pixel correspondant dans l'image qui nous sert de comparaison. Si c'est le cas, on incrémente le compteur. Après cela, il ne nous reste plus qu'à calculer la similitude en divisant le compteur par le nombre total de pixels.
Nous retournons ensuite la similitude.

Question 5 :

Nous devons écrire dans le fichier reconnaissance.py la fonction reconnaissance_chiffre(image, liste_modeles, S) qui va effectuer la reconnaissance de chiffre sur l'image image donnée en entrée.
Pour cela, il faudra dans la fonction tour à tour binariser l'image et la localiser. Nous calculerons ensuite sa similitude à tous les modèles et nous garderons le modèle ayant la plus grande similitude. Nous sauvegardons la similitude maximale et l'indice de l'image ayant la similitude maximale. Cette fonction renverra un entier compris entre 0 et 9.

Nous avons le code suivant :

```
15
16      def reconnaissance_chiffre(image, liste_modeles, S):
17          #initialisation du compteur de modèles :
18          compteur = 0
19          #initialisation du maximum de similitude :
20          similitude = 0
21          #binarisation et localisation de l'image étudiée :
22          image_bin = image.binarisation(S)
23          image_loc = image_bin.localisation()
24          #parcours des modèles :
25          for i in range(len(liste_modeles)):
26              modele = liste_modeles[i]
27              #redimensionnement de l'image pour la comparaison avec le modèle :
28              image = image_loc.resize(modele.H, modele.W)
29              temporaire = image.similitude(modele)
30              #comparaison de la similitude temporaire avec la similitude max :
31              if temporaire > similitude :
32                  #modification des compteurs et du max de similitude :
33                  similitude = temporaire
34                  compteur = i
35          #renvoi du numéro du modèle
36          return compteur
37
```

Explication :

On commence par initialiser un compteur afin de stocker le chiffre retenu, on initialise aussi une variable similitude pour stocker les similitudes les plus grandes au fur et à mesure de nos tests. Avant de commencer les tests, nous devons binariser et localiser notre image à l'aide de nos méthodes créées précédemment. On crée une boucle for ; pour chaque modèle de la liste, nous devons redimensionner notre image afin de pouvoir calculer la similitude entre les deux images, puis nous stockons cette similitude dans une variable temporaire. A l'aide d'une boucle if, nous testons si notre similitude temporaire est plus grande que celle retenue. Si c'est le cas, nous remplaçons notre variable similitude par notre variable temporaire et nous remplaçons notre compteur par le numéro du modèle.
Pour finir, nous retournons le compteur.

Nous obtenons le résultat suivant :

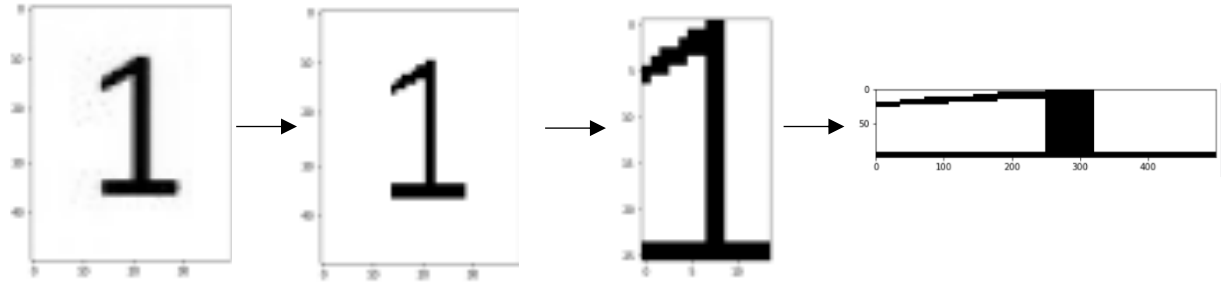


Image de base | Image binarisée | Image localisée | Image redimensionnée

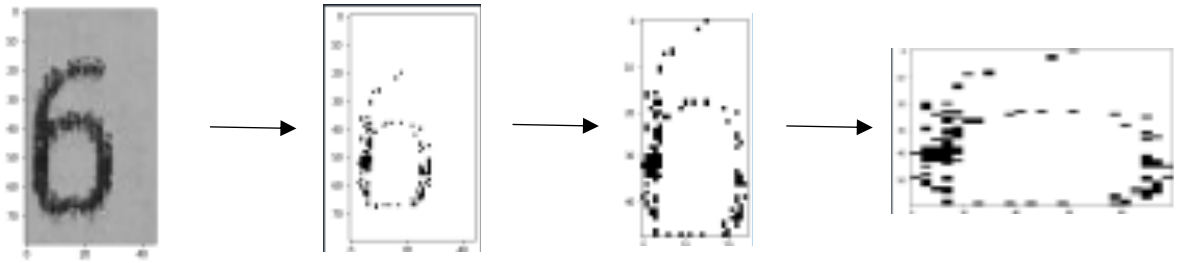
```
In [115]: runfile('/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-reconnaissance-chiffres-
tp2_dornier_leglise-main/src/main.py', wdir='/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-
reconnaissance-chiffres-tp2_dornier_leglise-main/src')
Reloaded modules: image, reconnaissance
lecture image : ../assets/test2.JPG (50x40)
lecture image : ../assets/_0.png (32x22)
lecture image : ../assets/_1.png (32x18)
lecture image : ../assets/_2.png (32x20)
lecture image : ../assets/_3.png (32x20)
lecture image : ../assets/_4.png (32x24)
lecture image : ../assets/_5.png (32x20)
lecture image : ../assets/_6.png (32x21)
lecture image : ../assets/_7.png (32x21)
lecture image : ../assets/_8.png (32x22)
lecture image : ../assets/_9.png (32x22)
Le chiffre reconnu est :  1

In [116]:
```

Après avoir effectué les tests, nous n'obtenons pas d'erreurs.

Question 6 :

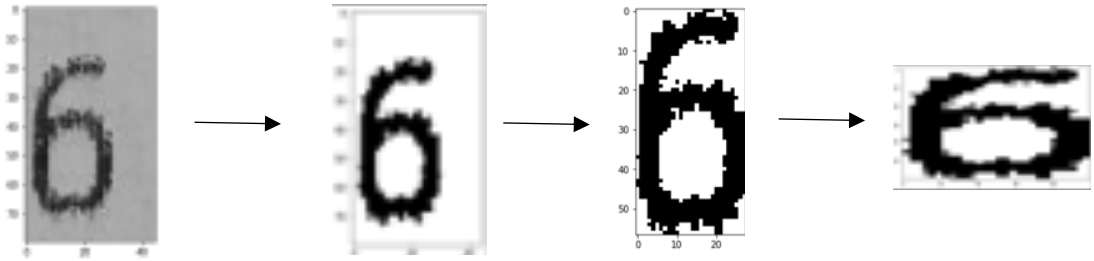
Nous modifions l'image de test et le seuil dans main.py afin de tester la fonction de reconnaissance. Nous testons avec plusieurs valeurs :

- S =40 et test10.jpg :



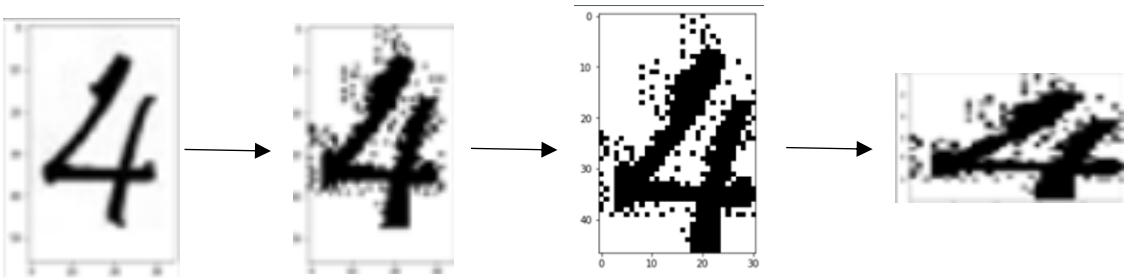
```
In [147]: runfile('/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-reconnaissance-chiffres-tp2_dornier_leglise-main/src/main.py', wdir='/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-reconnaissance-chiffres-tp2_dornier_leglise-main/src')
Reloaded modules: image, reconnaissance
lecture image : ../assets/test10.jpg (80x45)
lecture image : ../assets/_0.png (32x22)
lecture image : ../assets/_1.png (32x18)
lecture image : ../assets/_2.png (32x20)
lecture image : ../assets/_3.png (32x20)
lecture image : ../assets/_4.png (32x24)
lecture image : ../assets/_5.png (32x20)
lecture image : ../assets/_6.png (32x21)
lecture image : ../assets/_7.png (32x21)
lecture image : ../assets/_8.png (32x22)
lecture image : ../assets/_9.png (32x22)
Le chiffre reconnu est :  6
```

- S = 100 et test10.jpg :



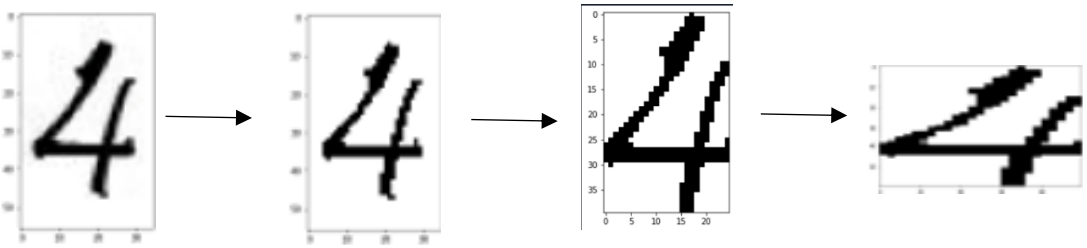
```
In [149]: runfile('/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-reconnaissance-chiffres-tp2_dornier_leglise-main/src/main.py', wdir='/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-reconnaissance-chiffres-tp2_dornier_leglise-main/src')
Reloaded modules: image, reconnaissance
lecture image : ../assets/test10.jpg (80x45)
lecture image : ../assets/_0.png (32x22)
lecture image : ../assets/_1.png (32x18)
lecture image : ../assets/_2.png (32x20)
lecture image : ../assets/_3.png (32x20)
lecture image : ../assets/_4.png (32x24)
lecture image : ../assets/_5.png (32x20)
lecture image : ../assets/_6.png (32x21)
lecture image : ../assets/_7.png (32x21)
lecture image : ../assets/_8.png (32x22)
lecture image : ../assets/_9.png (32x22)
Le chiffre reconnu est :  6
```

- S = 250 et test6.JPG :



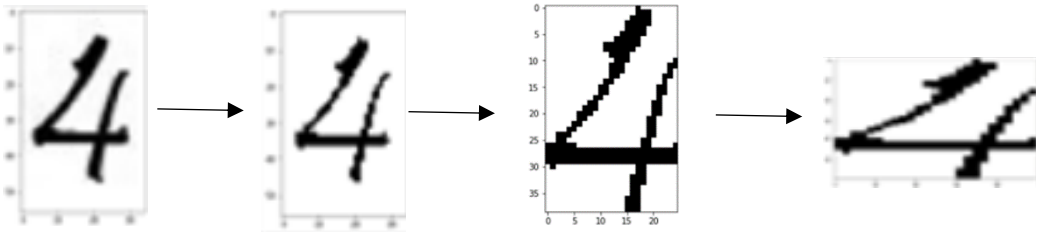
```
In [157]: runfile('/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-reconnaissance-chiffres-tp2_dornier_leglise-main/src/main.py', wdir='/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-reconnaissance-chiffres-tp2_dornier_leglise-main/src')
Reloaded modules: image, reconnaissance
lecture image : ../assets/test6.JPG (56x35)
lecture image : ../assets/_0.png (32x22)
lecture image : ../assets/_1.png (32x18)
lecture image : ../assets/_2.png (32x20)
lecture image : ../assets/_3.png (32x20)
lecture image : ../assets/_4.png (32x24)
lecture image : ../assets/_5.png (32x20)
lecture image : ../assets/_6.png (32x21)
lecture image : ../assets/_7.png (32x21)
lecture image : ../assets/_8.png (32x22)
lecture image : ../assets/_9.png (32x22)
Le chiffre reconnu est :  4
```

- S = 100 et test6.JPG :



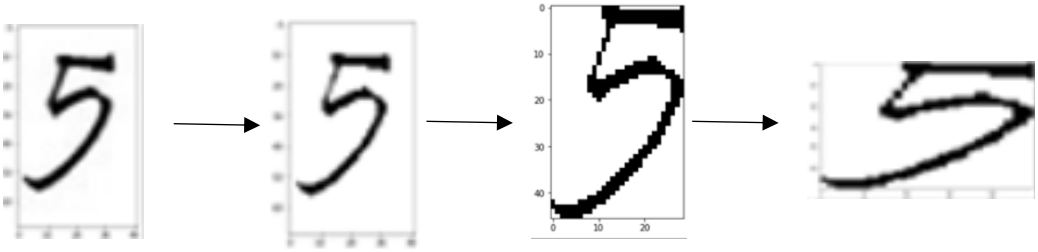
```
In [158]: runfile('/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-reconnaissance-chiffres-tp2_dornier_leglise-main/src/main.py', wdir='/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-reconnaissance-chiffres-tp2_dornier_leglise-main/src')
Reloaded modules: image, reconnaissance
lecture image : ../assets/test6.JPG (56x35)
lecture image : ../assets/_0.png (32x22)
lecture image : ../assets/_1.png (32x18)
lecture image : ../assets/_2.png (32x20)
lecture image : ../assets/_3.png (32x20)
lecture image : ../assets/_4.png (32x24)
lecture image : ../assets/_5.png (32x20)
lecture image : ../assets/_6.png (32x21)
lecture image : ../assets/_7.png (32x21)
lecture image : ../assets/_8.png (32x22)
lecture image : ../assets/_9.png (32x22)
Le chiffre reconnu est :  4
```

- S = 40 et test6.JPG :



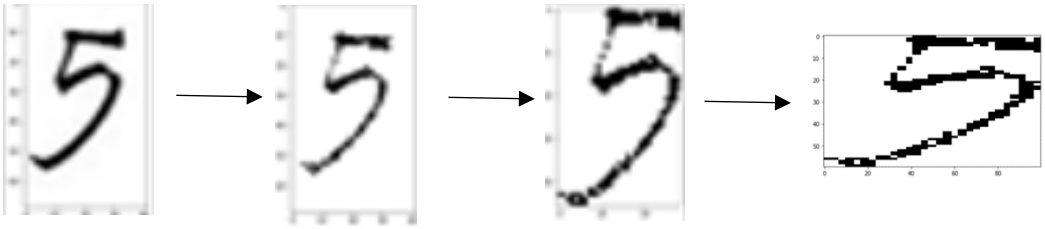
```
In [159]: runfile('/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-reconnaissance-chiffres-tp2_dornier_leglise-main/src/main.py', wdir='/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-reconnaissance-chiffres-tp2_dornier_leglise-main/src')
Reloaded modules: image, reconnaissance
lecture image : ../assets/test6.JPG (56x35)
lecture image : ../assets/_0.png (32x22)
lecture image : ../assets/_1.png (32x18)
lecture image : ../assets/_2.png (32x20)
lecture image : ../assets/_3.png (32x20)
lecture image : ../assets/_4.png (32x24)
lecture image : ../assets/_5.png (32x20)
lecture image : ../assets/_6.png (32x21)
lecture image : ../assets/_7.png (32x21)
lecture image : ../assets/_8.png (32x22)
lecture image : ../assets/_9.png (32x22)
Le chiffre reconnu est :  4
```

- S = 40 et test7.JPG :



```
In [160]: runfile('/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-reconnaissance-chiffres-tp2_dornier_leglise-main/src/main.py', wdir='/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-reconnaissance-chiffres-tp2_dornier_leglise-main/src')
Reloaded modules: image, reconnaissance
lecture image : ../assets/test7.JPG (69x42)
lecture image : ../assets/_0.png (32x22)
lecture image : ../assets/_1.png (32x18)
lecture image : ../assets/_2.png (32x20)
lecture image : ../assets/_3.png (32x20)
lecture image : ../assets/_4.png (32x24)
lecture image : ../assets/_5.png (32x20)
lecture image : ../assets/_6.png (32x21)
lecture image : ../assets/_7.png (32x21)
lecture image : ../assets/_8.png (32x22)
lecture image : ../assets/_9.png (32x22)
Le chiffre reconnu est :  5
```

- S = 10 et test7.JPG :



```
In [163]: runfile('/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-reconnaissance-chiffres-tp2_dornier_leglise-main/src/main.py', wdir='/Users/cloe/Documents/AAPolytech/SNI3/INFO501/TP/tp2-reconnaissance-chiffres-tp2_dornier_leglise-main/src')
Reloaded modules: image, reconnaissance
lecture image : ../assets/test7.JPG (69x42)
lecture image : ../assets/_0.png (32x22)
lecture image : ../assets/_1.png (32x18)
lecture image : ../assets/_2.png (32x20)
lecture image : ../assets/_3.png (32x20)
lecture image : ../assets/_4.png (32x24)
lecture image : ../assets/_5.png (32x20)
lecture image : ../assets/_6.png (32x21)
lecture image : ../assets/_7.png (32x21)
lecture image : ../assets/_8.png (32x22)
lecture image : ../assets/_9.png (32x22)
Le chiffre reconnu est :  5
```

La valeur de seuil qui marche le mieux selon nos expérimentations est 100, il faut prendre un seuil ni trop grand ni trop petit.

Remarque :

- Les images comportant plusieurs chiffres faussent les résultats.
- Même avec des valeurs de seuil très élevées ou très petites, la reconnaissance arrive à se faire.