

## Rapport de TP2 – Lecture automatique de chiffre par analyse d'image

### II. Prise en main de l'environnement

Pour commencer nous avons commenté le programme main à partir de la ligne 30.

The screenshot displays the Spyder Python IDE interface. The left pane shows a Python script named `main.py` with the following content:

```
1 '''
2 File: main.py
3 Created Date: Friday August 27th 2021 - 02:35pm
4 Author: Ammar Nian
5 Contact: ammar_nian@univ-smb.fr
6 -----
7 Last Modified: Mon Aug 30 2021
8 Modified By: Ammar Nian
9 -----
10 Copyright (c) 2021 Université Savoie Mont-Blanc
11 '''
12 import matplotlib.pyplot as plt
13 from image import image
14 from reconnaissance import reconnaissance_chiffre, lecture_modeles
15
16
17 if __name__ == '__main__':
18
19     # Variables utiles
20     path_to_assets = '../assets/'
21     plt.ion() # Mode interactif de matplotlib pour ne pas bloquer l'exécution lorsque l'on fait display
22
23     # Lecture image et affichage
24     # =====
25     image = image()
26     image.load(path_to_assets + 'test2.JPG')
27     image.display("Exemple d'image")
28
29     # Binarisation de l'image et affichage
30     # =====
31     S = 70
32     image_binarisee = image.binarisation(S)
33     image_binarisee.display("Image binarisee")
34
35     # Localisation de l'image et affichage
36     # =====
37     image_localisee = image_binarisee.localisation()
38     image_localisee.display("Image localisee")
39
40     # Redimensionnement de l'image et affichage
41     # =====
42     image_redimensionnee = image_localisee.resize(100, 500)
43     image_redimensionnee.display("Image redimensionnee")
44
45     # Lecture modeles et reconnaissance
46     # =====
47     liste_modeles = lecture_modeles(path_to_assets)
48     chiffre = reconnaissance_chiffre(image_redimensionnee, liste_modeles, 70)
49     print("Le chiffre reconnu est : ", chiffre)
```

The right pane shows the execution results. The top part displays a plot of the digit '1' with axes ranging from 0 to 40. The bottom part shows the console output, which includes a traceback error:

```
Traceback (most recent call last):
  File "C:\Users\Daniel\Downloads\tp2-reconnaissance-chiffres-henon-calvin-paris-daniel-tp2-main\src\main.py", line 35, in <module>
    image_binarisee.display("Image binarisee")
AttributeError: 'NoneType' object has no attribute 'display'

In [2]:
```

Lorsqu'on lance le programme une erreur apparait dans la console, malgré tout une image de 1 s'affiche

### III. Travail préparatoire

1):

Le **H** (height) signifie le nombre de pixels en hauteur dans l'image et **W** (width) signifie le nombre de pixels dans la largeur de l'image.

2) Binarisation de l'image :

Le but de cette étape est de convertir chaque pixel d'une image codée sur une base de 256 valeurs à une nouvelle base de 2 valeurs. Pour réaliser ceci, on complète le code fourni avec la question.

Code :

```
def binarisation(self, S):  
  
    im_bin = Image()  
  
    im_bin.set_pixels(np.zeros((self.H, self.W), dtype=np.uint8))  
  
    for i in range(0, self.H, 1):  
        for j in range(0, self.W, 1):  
            if self.pixels[i][j] >= S:  
                im_bin.pixels[i][j] = 255  
            else:  
                im_bin.pixels[i][j] = 0  
    return im_bin
```

Résultat :

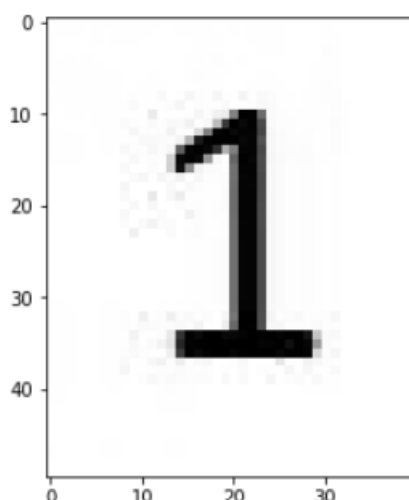


Image de base

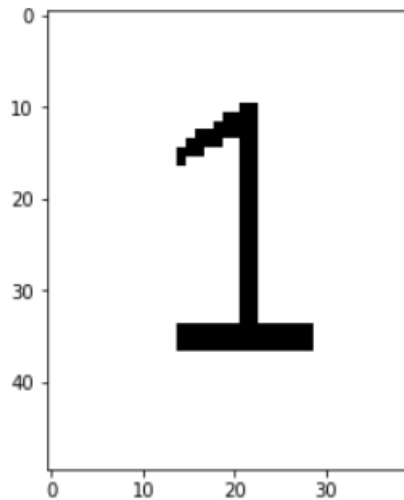


Image après binarisation

Après vérification avec le test image, aucune erreur est détectée et comme nous montre le résultat ci-dessus, le programme de binarisation fonctionne.

3) Localisation :

Le but de cette étape est d'ajuster le cadre de l'image en fonction de la taille de l'image (du nombre de pixel de celle-ci).

Code :

```
def localisation(self):
    im_loc = Image()
    lmin = self.H-1
    lmax = 0
    cmin = self.W-1
    cmax = 0
    for l in range(self.H):
        for c in range(self.W):

            if l < lmin:
                if self.pixels[l][c]==0:
                    lmin = l
            if l >= lmax:
                if self.pixels[l][c]==0:
                    lmax = l
            if c < cmin:
                if self.pixels[l][c]==0:
                    cmin = c

            if c >= cmax:
                if self.pixels[l][c]==0:
                    cmax = c

    im_loc.set_pixels(self.pixels[lmin:lmax,cmin:cmax])
    return im_loc
```

Résultat :

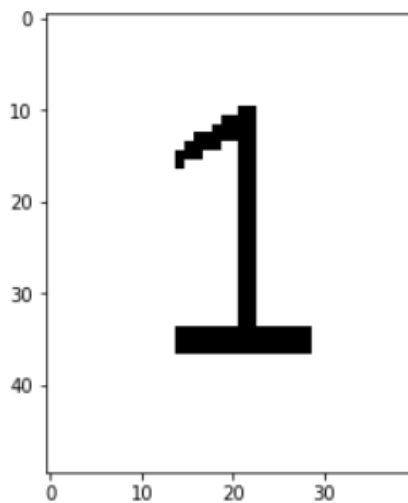


Image après binarisation

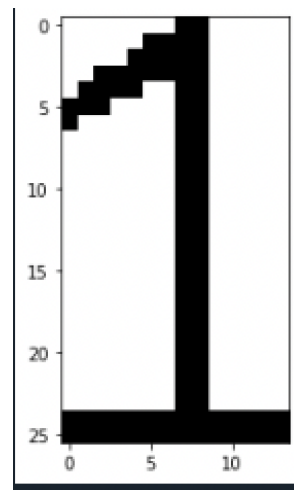


Image après localisation

Ainsi on peut voir qu'après l'application du programme de la localisation, le cadre de l'image est recadré en fonction de la taille de celle-ci. Le programme de localisation fonctionne.

#### IV. Reconnaissance automatique de chiffre

1) Pour cette question, nous devons changer la valeur de seuil de la binarisation de l'image dans le fichier main.py.

Valeur initiale du seuil de la binarisation

```
S = 70  
image_binarisee = image.binarisation(S)  
image_binarisee.display("Image binarisee")
```

Image pour un seuil de 30 :

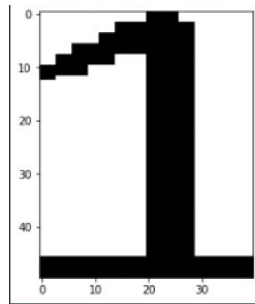


Image pour un seuil de 60 :

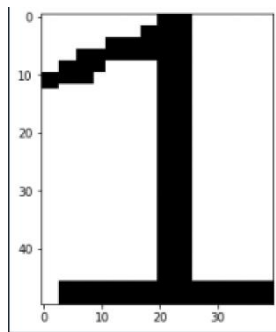
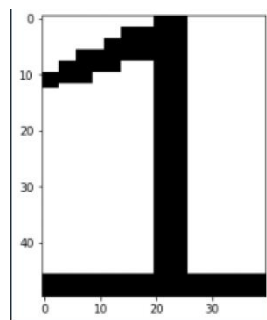


Image pour un seuil de 90 :



On lance le test\_image.py et aucune erreur n'est détectée.

2)

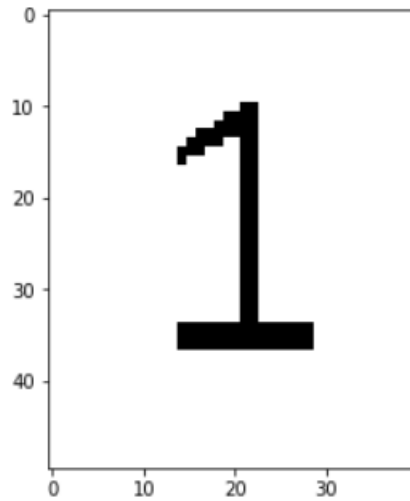


Image après binarisation

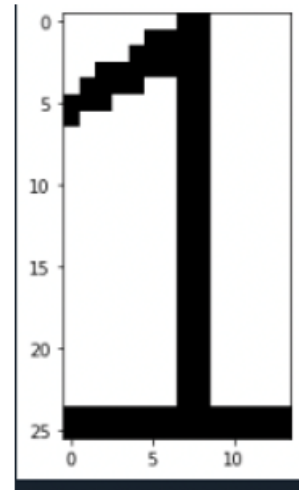


Image après localisation

Comme montré précédemment, voici le résultat de la méthode de localisation.

3) Pour cette question nous devons ajouter à la classe image la méthode resize qui redimensionne l'image à la taille voulue.

Code :

```
def resize(self, new_H, new_W):
    im_bin=Image()
    tab=resize(self.pixels, (new_H,new_W),0)
    tab_bin=(np.uint8(tab*255))
    im_bin.set_pixels(tab_bin)
    return im_bin
```

Pour cette question, nous devons changer les valeurs du dimensionnement de l'image dans le fichier main.py.

Valeur des dimensions initiales :

```
image_resizee = image_localisee.resize(50, 40)
image_resizee.display("Image redimensionnee")
```

Image pour une dimension de (130 ,80)

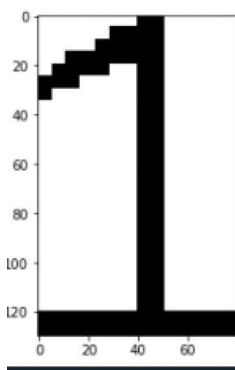


Image pour une dimension de (30,15)

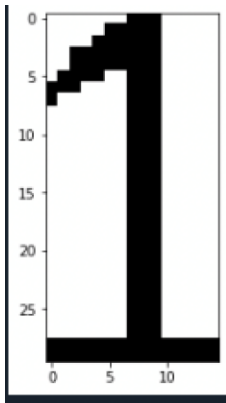
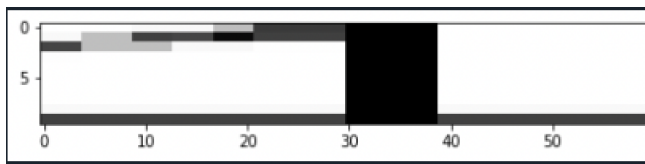


Image pour une dimension de (10,60)



4) Pour cette question, on applique l'explication fournie à l'étape 4 de la partie 1 du TP qui nous permet d'appliquer à la classe image la méthode similitude.

Code :

```
def similitude(self, im):
    n=0
    for l in range(self.H):
        for c in range(self.W):
            if self.pixels[l][c]==im.pixels[l][c]:
                n+=1
    return n/(self.H*self.W)
```

La fonction similitude permet de comparer 2 images afin de déterminer leur niveau de ressemblance. Deux images totalement différentes donneront 0 alors que 2 images parfaitement identiques donneront 1.

5) Dans cette question nous devons écrire la fonction reconnaissance\_chiffre qui permet de reconnaître un chiffre. Pour se faire on reprend les mêmes fonctions que vu précédemment (binarisation, localisation).

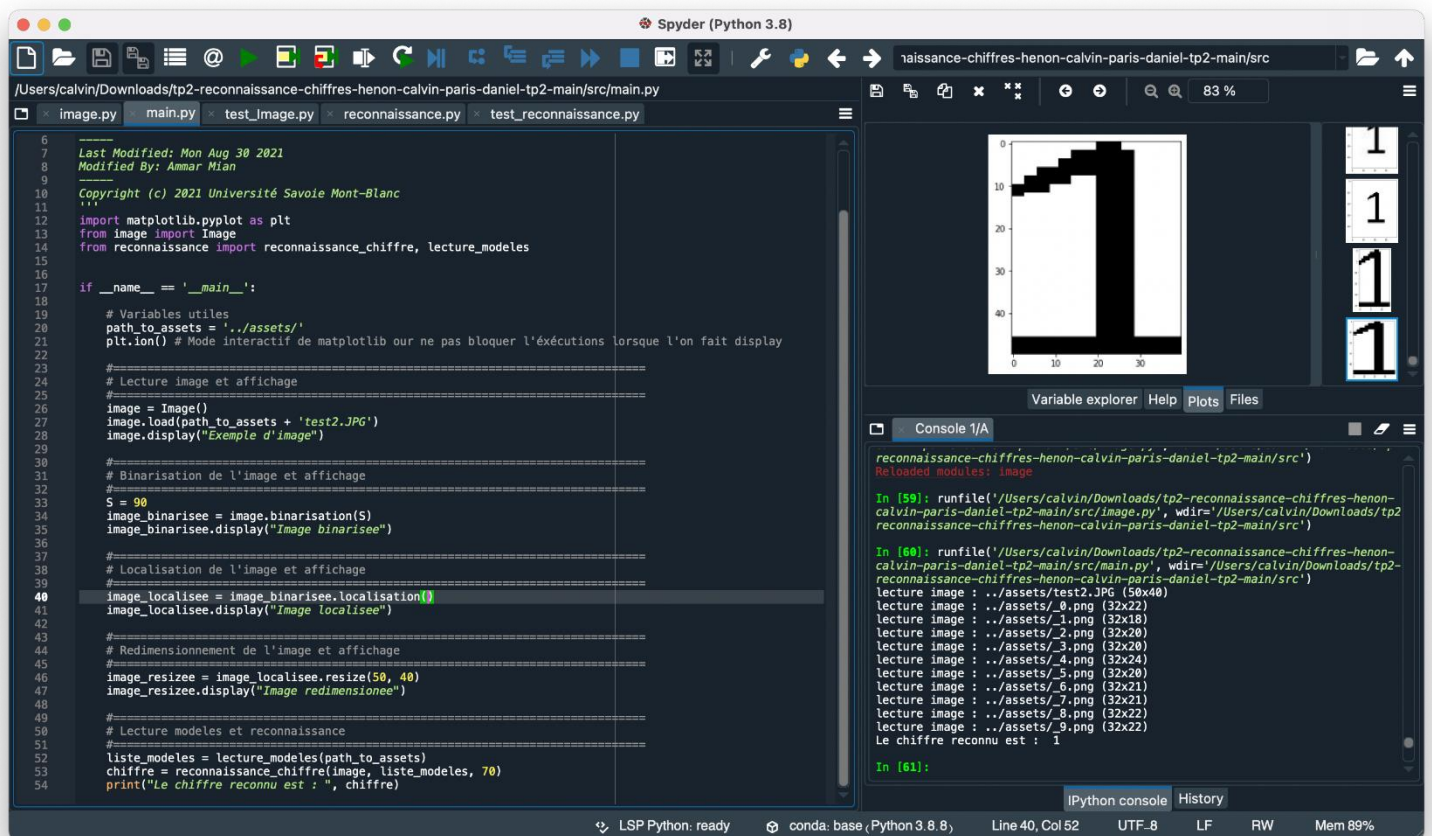
Code :

```
from image import Image

def lecture_modeles(chemin_dossier):
    fichiers= ['_0.png', '_1.png', '_2.png', '_3.png', '_4.png', '_5.png', '_6.png',
               '_7.png', '_8.png', '_9.png']
    liste_modeles = []
    for fichier in fichiers:
        model = Image()
        model.load(chemin_dossier + fichier)
        liste_modeles.append(model)
    return liste_modeles

def reconnaissance_chiffre(image, liste_modeles, S):
    image=image.binarisation(S)
    image=image.localisation()
    similitude=0
    for i in range(len(liste_modeles)):
        im=image.resize(liste_modeles[i].H, liste_modeles[i].W)
        if im.similitude(liste_modeles[i])>similitude:
            similitude=im.similitude(liste_modeles[i])
            indiceim=i
    return indiceim
```

Résultat : on exécute le main.py et on obtient le résultat suivant :



```
6
7 Last Modified: Mon Aug 30 2021
8 Modified By: Ammar Mian
9
10 Copyright (c) 2021 Université Savoie Mont-Blanc
11
12 import matplotlib.pyplot as plt
13 from image import Image
14 from reconnaissance import reconnaissance_chiffre, lecture_modeles
15
16
17 if __name__ == '__main__':
18
19     # Variables utiles
20     path_to_assets = '../assets/'
21     plt.ion() # Mode interactif de matplotlib pour ne pas bloquer l'exécution lorsque l'on fait display
22
23     # Lecture image et affichage
24     #
25     image = Image()
26     image.load(path_to_assets + 'test2.JPG')
27     image.display("Exemple d'image")
28
29     #
30     # Binarisation de l'image et affichage
31     #
32     S = 90
33     image_binarisee = image.binarisation(S)
34     image_binarisee.display("Image binarisee")
35
36     #
37     # Localisation de l'image et affichage
38     #
39     image_localisee = image_binarisee.localisation()
40     image_localisee.display("Image localisee")
41
42     #
43     # Redimensionnement de l'image et affichage
44     #
45     image_resizee = image_localisee.resize(50, 40)
46     image_resizee.display("Image redimensionnee")
47
48     #
49     # Lecture modeles et reconnaissance
50     #
51     liste_modeles = lecture_modeles(path_to_assets)
52     chiffre = reconnaissance_chiffre(image, liste_modeles, 70)
53     print("Le chiffre reconnu est : ", chiffre)
54
```

reconnaissance-chiffres-henon-calvin-paris-daniel-tp2-main/src

Variable explorer Help Plots Files

Console 1/A

```
reconnaissance-chiffres-henon-calvin-paris-daniel-tp2-main/src')
Reloaded modules: image

In [59]: runfile('/Users/calvin/Downloads/tp2-reconnaissance-chiffres-henon-calvin-paris-daniel-tp2-main/src/image.py', wdir='/Users/calvin/Downloads/tp2-reconnaissance-chiffres-henon-calvin-paris-daniel-tp2-main/src')

In [60]: runfile('/Users/calvin/Downloads/tp2-reconnaissance-chiffres-henon-calvin-paris-daniel-tp2-main/src/main.py', wdir='/Users/calvin/Downloads/tp2-reconnaissance-chiffres-henon-calvin-paris-daniel-tp2-main/src')
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

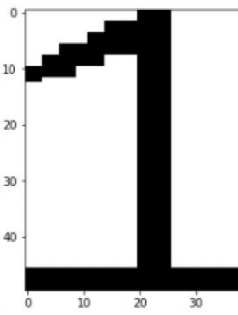
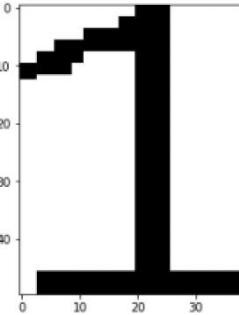
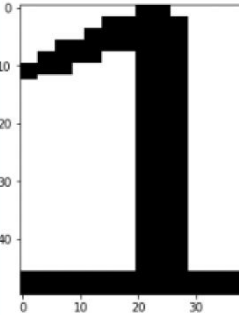
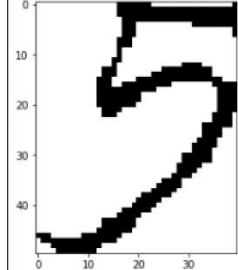
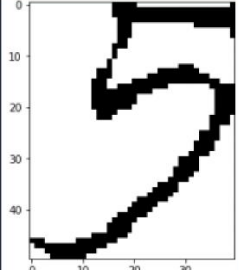
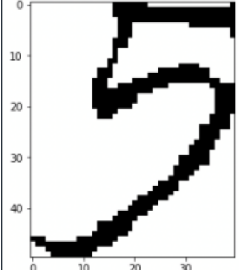
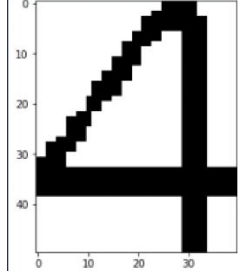
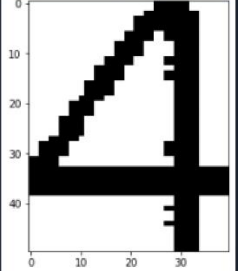
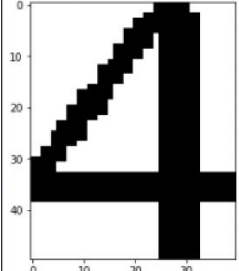
In [61]:
```

IPython console History

LSP Python: ready conda: base (Python 3.8.8) Line 40, Col 52 UTF-8 LF RW Mem 89%

Le chiffre détecté est le 1 ce qui est le bon résultat, de plus on lance le fichier de test test\_reconnaissance.py. Aucune erreur détectée.

6)

Nom du fichier	Seuil=30	Seuil=60	Seuil=90	Résultat
Test2.jpeg				<pre>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</pre>
Test7.jpeg				<pre>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</pre>
Test1.jpeg				<pre>lecture image : ../assets/test1.JPG (54x40) 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</pre>

Nos expérimentations nous ont amené à préférer le seuil de 90 par rapport au seuil 30 et 60 car il permet d'obtenir un chiffre mieux représenté.