PACCOUD William

RISS Ryan

19/11/2021

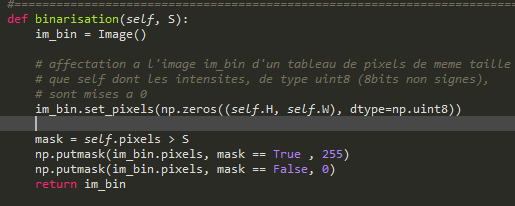
# Rapport de TP2 - Lecture automatique de chiffres par analyse d’image

## Introduction

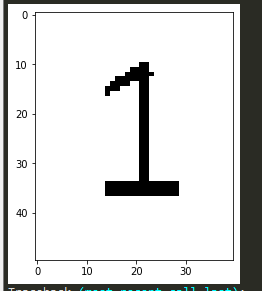
Ce TP porte sur la reconnaissance de chiffre. Celle-ci va s’opérer en 5 étapes : binarisation, localisation, adaptation de la taille, mesure de ressemblance et décision, que nous allons devoir implémenter.

## Travail préparatoire

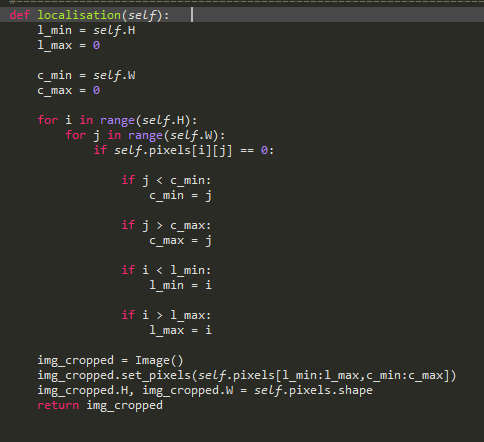
### Méthode binarisation



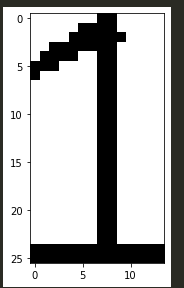
Problème : il faut binariser une image celon un certain seuil. Pour ce faire j’ai créer un masque correspondant aux valeurs de l’image supérieur au seuil. J’applique ensuite ce masque avec la fonction **putmask** pour modifier notre nouvelle image à binariser celon les valeurs booléennes du masque pour binariser l’image.



### Localisation



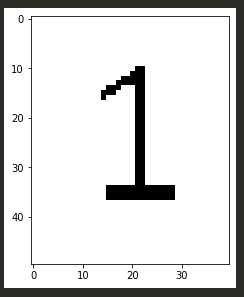
Problème : Il faut redimensionner l’image pour qu’elle ne contienne que le chiffre. Il faut donc trouver les coordonnées minimales et maximales du rectangle qui englobent le chiffre. Pour ce faire on parcourt l’image jusqu’à rencontrer un pixel noir et on vérifie si ses coordonnées sont un maximum ou un minimum.



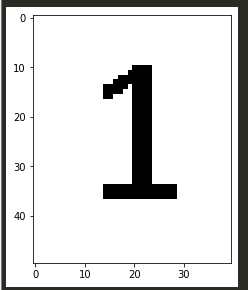
## Section 2 du TP

### Test Seuils

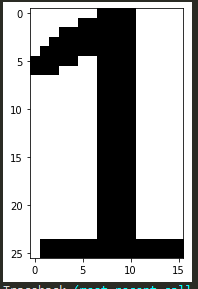
S = 20



S= 150

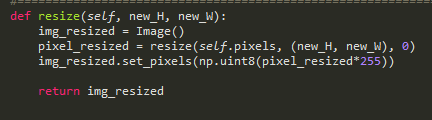


S = 220

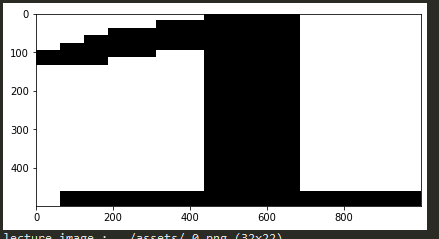


On observe que plus le seuil est petit (et donc la tolérance plus petite), plus la forme du chiffre s’affine car l’image de base contient beaucoup de pixels moins foncé au niveau de la barre diagonale du 1.

### Méthode Resize

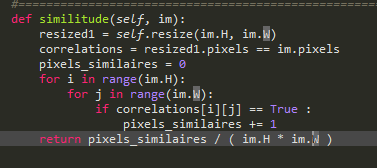


Problème : redimmensionner une image. On utilise la fonction image\_resize donnée et on reconverti les valeurs en uint8.



Exemple avec (500,1000)

### Similitude



Problème : on veut mesurer la similitude entre deux images. Nous avons procédé de la manière suivante : on redimensionne l’image à la taille du modèle, puis on obtient un masque qu’on appelle « correlations ». Ce masque est de la même forme que les images et contient True comme valeur aux coordonnées des pixels similaires. Ensuite on compte le nombre de True dans ce tableau pour avoir le nombre de pixels similaires. Puis on divise le nombre de pixels similaires par le nombre de pixels totaux pour obtenir une valeur de similitude entre 0 et 1.

PROBLEME :

Nous rencontrons un cas incongru… lorsque nous comparons une image à elle-même, nous n’obtenons pas un taux de similitude de 1 mais quelque chose comme 0.99. Après avoir effectué des tests, on se rend compte que cela vient de la fonction **resize** : en effet, si on l’a compare à elle-même, l’image est passé dans la fonction **resize** avec pour argument ses propres dimensions, et cela semble la modifier quand même de quelques pixels :





On voit bien que notre fonction compte 1622 pixels similaires pour 1634 totaux…

Sans **resize** :

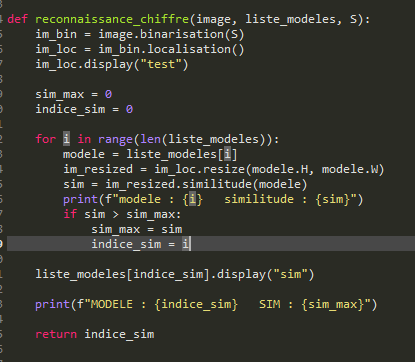




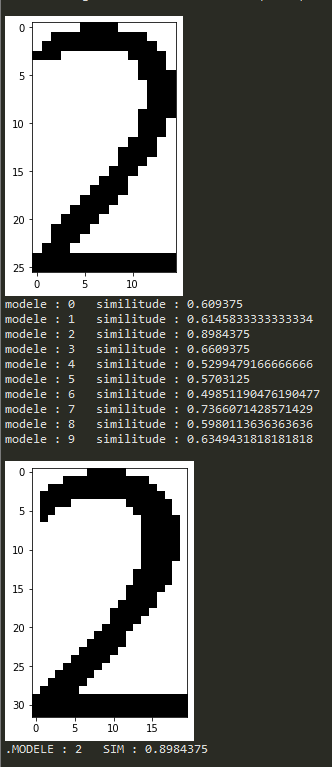
Sans redimensionner l’image par elle-même la fonction **similitude** marche bien…

Nous ne savons pas d’où vient cette anomalie, notre fonction **resize** a bien été validé plusieurs fois par l’enseignant.

### Reconnaissance



Le problème : trouver l’image la plus similaire parmi un ensemble de modèle. Notre solution : on commence par binariser et localiser l’image. Ensuite, pour chaque modele dans la liste, on redimensionne l’image à la taille du modèle et on calcule leur similitude pour trouver la plus grande similitude.



### Tests différents seuils/modèles

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 250 | 180 | 100 | 50 |
| Test6.jpg ( 4 stylisé) | 4 -0,68 | 4 - 0,71 | 4 - 0.71 | 4 – 0.73 |
| Test9.jpg ( plusieurs chiffre) | 4 – 0.55 | 7 – 0.64 | 7 – 0.66 | 7 – 0.67 |
| Test10.jpg (6 mauvaise qualité) | 8 – 0.46 | 8 – 0.46 | 6 – 0.77 | 6 – 0.66 |
| Test3.jpg (2 décalé) | 4 – 0.58 | 2 – 0.87 | 2 – 0.89 | 2 – 0.89 |

D’après ces observations, il semblerait que la méthode a du mal a donner des resultats cohérents quand l’image est trop différente des modèles ou le chiffre mal discernable. Lorsqu’il s’agit d’un chiffre bien délimité, un seuil entre 100 et 50 semble être la meilleure option pour obtenir le bon résultat avec le meilleur taux de similitude.

## Conclusion

En conclusion, ce TP fut très interessant, mis à part la petite anomalie de la fonction similitude décrite plus haut. Autrement notre avancement fut fluide. Ce fut intéressant de réaliser une tâche qui parait plus compliqué que ce dont on pourrait penser ( en tout cas à ce niveau de complexité ). Quand on pense reconnaissance de chiffres on pense souvent IA et neural networks mais moins souvent que c’est aussi réalisable en programmation « classique ».