Mini projet :
Adaptive document image binarization – Text binarization

Introduction

Ce « mini-projet » a pour but de programmer la méthode de binarisation locale de texte TBM (Text binarization method) développé par Jaakko SAUVOLA en 1999. Celle-ci se base sur le principe de binarisation de la méthode de Niblacks, et l'améliore en introduisant de nouveaux termes dans le calcul de la valeur du threshold. Le résultat obtenu est ainsi bien meilleur lorsque les changements de luminosités spatiales sur l'image sont très forts.

I) Principe de L'algorithme de Sauvola

L'idée de l'algorithme est d'obtenir une valeur locale de threshold pour chaque pixel de l'image. Cette valeur va être calculée à partir de la moyenne *m* et de l'écart type *s* des plus-proches voisins de chaque pixel. La formule de calcul est la suivant :

$$Threshold(x,y,r) = m(x,y,r) * \left[1 + k * \left(\frac{s(x,y,r)}{R} - 1\right)\right]$$

m(x,y,r) : moyenne calculé pour le pixel (x,y) avec les plus-proches voisins situées à une distance r
s(x,y,r) : écart type calculé pour le pixel (x,y) avec les plus-proches voisins situées à une distance r
M : Constante - plage dynamique de l'écart type fixée à 128 (50%)
k : Constante positive fixée à 0.5

Par rapport à la formule de Niblack (*Threshold = m + k*s, k<0*), l'écart type a une contribution plus forte dans le résultat. Si le fond du texte est très foncé, la faible valeur de *m* fait baisser le threshold. S'il est très clair, m va la faire augmenter. Le terme avec s permet d'obtenir une valeur plus adaptée à chaque pixel. Ainsi, la binarisation est beaucoup plus propre avec cette méthode qu'avec celle de Sauvola car on supprime le bruit de pixels noirs présent en arrière-plan.

Dans l'algorithme proposé, les valeurs de threshold sont seulement calculées pour certains pixels appelées pixels de base. Ce sont les P(x,y) avec x et y multiples d'un entier *n*. Ensuite les valeurs de threshold des autres pixels sont obtenues par interpolation (simple ou bilinéaire). Enfin, la valeur de chaque pixel est comparée à sa valeur de threshold afin d'être classifiées en noir ou blanc .

$$\begin{cases} Si\ P(x,y) > Threshold(x,y,r), & Bin(x,y) = 1 \quad (pixel\ blanc) \\ Sinon & Bin(x,y) = 0 \quad (pixel\ noir) \end{cases}$$

II) Méthodes d'interpolation utilisées

a. *Interpolation simple*

Dans ce choix d'interpolation, les valeurs des pixels interpolés sont calculées comme étant la moyenne des pixels de base les plus proches. On distingue 4 cas décris par les schémas suivants (*figure 1*) suivant que le pixel soit sur un bord ou non. Ta, Tb, Tc et Td sont les valeurs de threshold calculées grâce à l'algorithme.

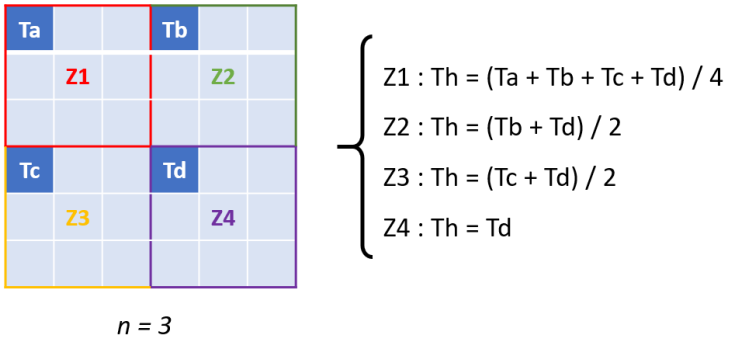


Figure 1: Interpolation simple pour des pixels dans les 4 situations possibles (zone 1 à 4)

b. Interpolation bilinéaire

Dans cette méthode, la fonction Th(x,y) est modélisée par une forme quadratique du type : *ax + by + cxy + d*. L'interpolation bilinéaire offre un résultat plus précis, particulièrement lorsque n>5. On retrouve aussi 4 cas différents (*figure 2*) . Les terme (k,l) représentent la distance du pixel par rapport à son plus proche pixel supérieur gauche calculée par l'algorithme.

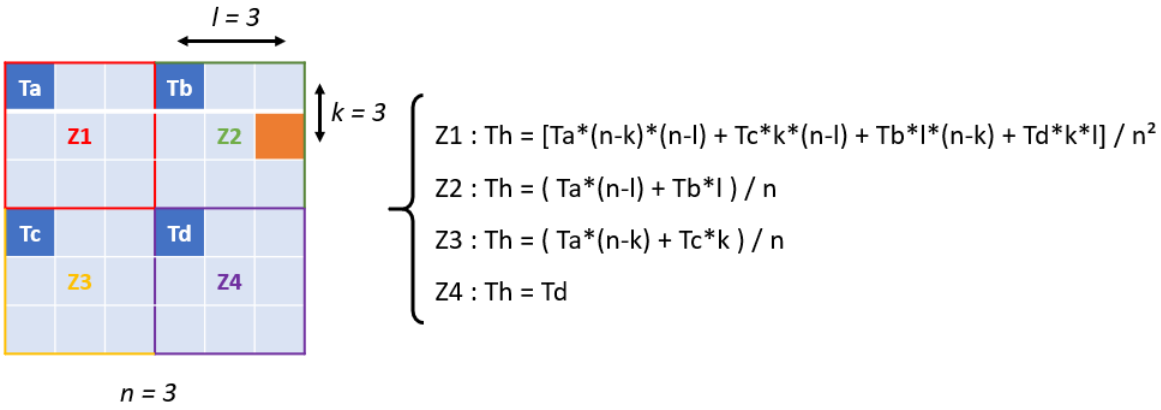


Figure 2: Interpolation bilinéaire pour les pixels dans les 4 situations possibles.
Exemple de la valeur de (k,l) pour le pixel orange

III) Programmation de la méthode

Le programme principal *Sauvola.py* contient les fonctions liées à la méthode de Sauvola. Le fichier *affiche.py* contient des fonctions pour importer et afficher les images.

Les fonctions de la méthode Sauvola sont :

- Sauvola(image, n, r, R = 128, k = 0.5, interpolation = "SIMPLE") la fonction principale de la méthode.
- interpol_bili(n, image, Threshold) et interpol_simple(n, Threshold) contenant les méthodes d'interpolations.
- Sauv_mean(image, x,y,r) et Sauv_standard_deviation(image, x, y, r, Mean) utilisées pour calculer la moyenne et l'écart type liée à chaque pixel (x,y).

IV) Résultats

Afin de voir l'influence des paramètres n et r, je réalise des premiers tests sur une image basique de taille (118x93) . Je prends **R = 128** et **k = 0.5,** comme suggéré dans l'article scientifique.

a. Influence de r

En premier lieu, je fais varier r avec les valeurs **r = 1,3,5,10,15,20**. On remarque qu'augmenter r augmente rapidement le temps de calcul de l'algorithme. On observe « à vue d'œil » que la qualité de la binarisation augmente atteint un maximum et stagne pour légèrement baisser. La meilleure qualité serait entre r = 5 et r = 10 .

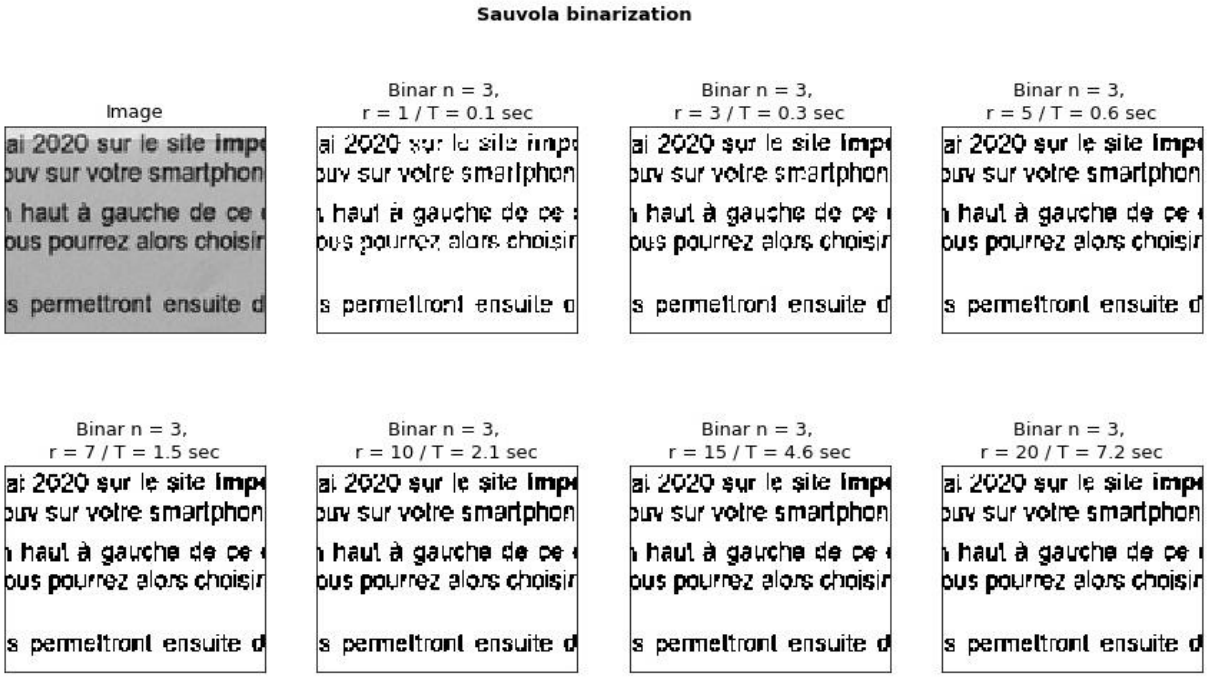


Figure 3: Sauvola binarisation avec différents r

b. Influence de n

Ensuite je fais varier n sur le même échantillon. Je fixe le nombre **r = 7**, des valeurs de **n = 1,3,5,7,10,15,20**. Je teste aussi les deux méthodes d'interpolation. On remarque que même si on augmente beaucoup n, on ne perd pas tellement en précision et on gagne énormément en temps de calcul. Pour les grosses valeurs de n, il semble que l'interpolation bilinéaire fait un peu mieux que l'interpolation linéaire même si les différences sont faibles.

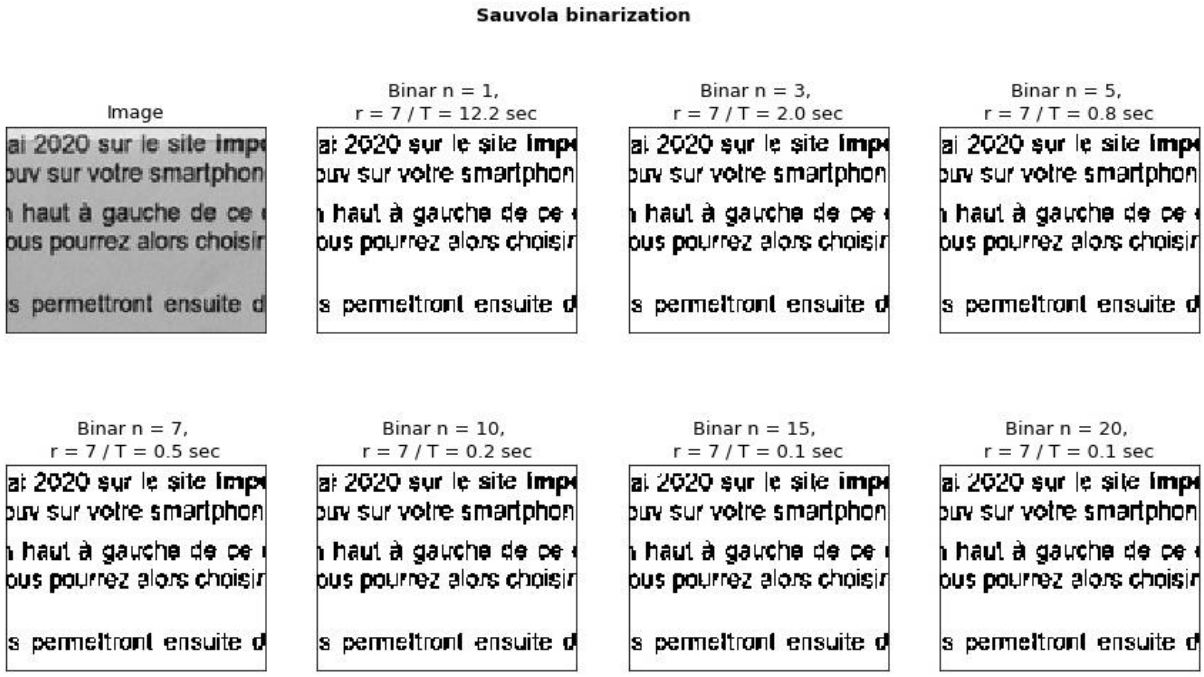


Figure 4: Binarisation avec différentes valeurs de n. Interpolation Linéaire

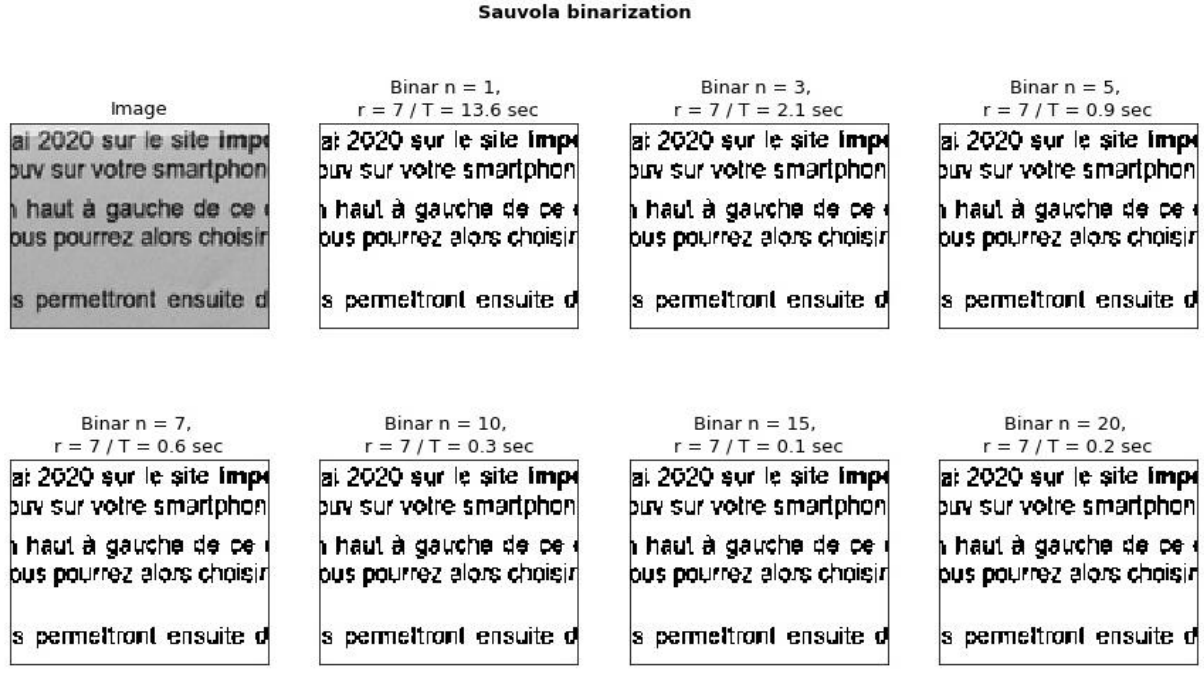
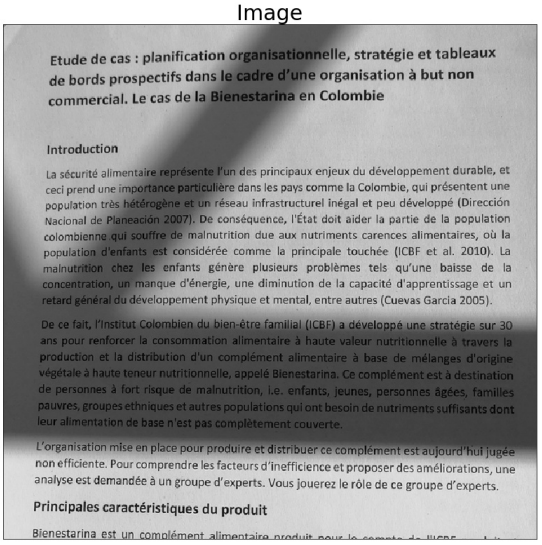


Figure 5: Binarisation avec différentes valeurs de n. Interpolation bilinéaire

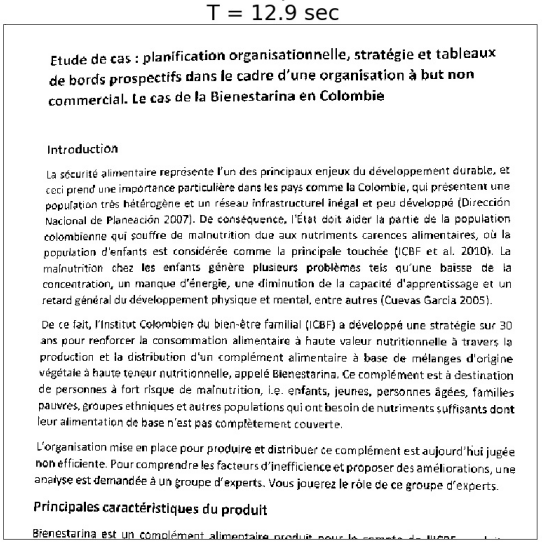
c. Test sur une image avec un fort changement des luminosités

Je teste ensuite les performances de l'algorithme sur un image de plus grande taille (641x618). J'utilise l'algorithme de Sauvola (**n = 5** avec les 2 méthodes d'interpolation puis utilise la méthode de Niblack avec interpolation Bilinéaire (**k = -0.2**). On remarque alors que la méthode de Sauvola obtient de bien meilleurs résultats que la méthode de Niblack qui réagit bizarrement aux changements de luminosité. Enfin, l'interpolation bilinéaire nous permet d'avoir un résultat légèrement plus doux à la lecture.

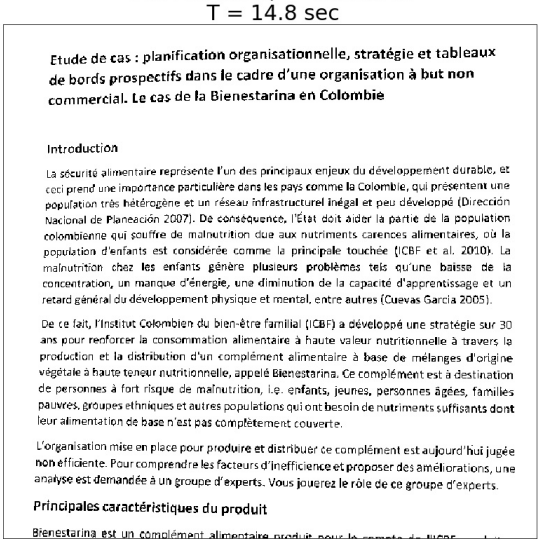
Image



Sauvola interp = SIMPLE
T = 12.9 sec



Sauvola interp = BILINEAR
T = 14.8 sec



Niblack interp = BILINEAR
T = 13.0 sec

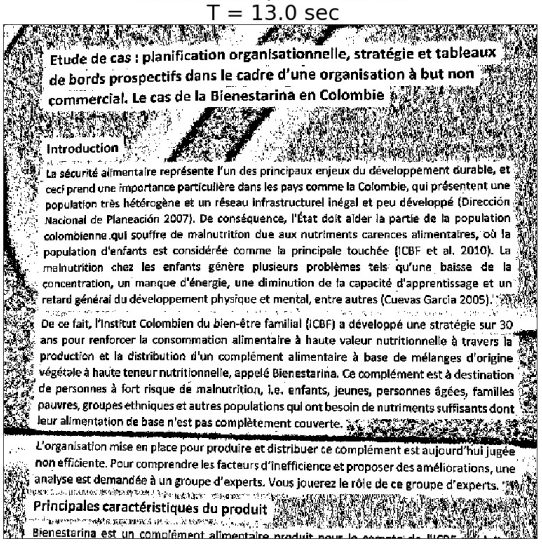


Figure 6: Résultats des algorithmes de Sauvola et Niblack

V) Conclusion

On remarque que l'algorithme de Sauvola remplit bien son rôle et apporte une véritable amélioration à l'algorithme de Niblack. Les images binarisées sont très précises malgré le bruit et les variations de luminosité présentes sur les photos.