Université Sidi Mohamed Ben Abdellah

Faculté Des Sciences Et Techniques Fès
Département d'Informatique

Projet de Fin d'Études

Master Sciences et Techniques
Systèmes Intelligents & Réseaux

Contribution à la segmentation du texte arabe imprimé

Lieu de stage : Laboratoire des Systèmes Intelligents et Applications (LSIA FSTF)

Réalisé par : ZOIZOU Abdelhay

Soutenu le 16/06/2017

Encadré par :

Pr. Ilham CHAKER
Pr. Arsalane ZARGHILI

Devant le jury composé de :

Pr. Ilham CHAKER
Pr. Arsalane ZARGHILI
Pr. Jamal KHARROUBI
Pr. Khalid ABBAD
Pr. Rachid BEN ABBOU

Année Universitaire : 2016 - 2017


بسم الله الرحمن الرحيم

Dédicaces

A ma très chère mère

Autant de phrases aussi expressives soient-elles ne sauraient montrer le degré d'amour et d'affection que j'éprouve pour toi. Tu n'as cessé de me soutenir et de m'encourager durant toutes les années de mes études. En ce jour mémorable, pour moi ainsi que pour toi, reçoit ce travail en signe de ma vive gratitude et ma profonde estime. Puisse le tout puissant te donner santé, bonheur et longue vie afin que je puisse te combler à mon tour.

A mon très chère père

Autant de phrases et d'expressions aussi éloquentes soient-elles ne sauraient exprimer ma gratitude et ma reconnaissance. Tu as su me faire apprendre le sens de la responsabilité, de l'optimisme et de la confiance en soi face aux difficultés. Tu as toujours guidé mes pas vers la réussite. Je te dois ce que je suis aujourd'hui et ce que je serai. Que Dieu te préserve, t'accorde santé, bonheur, et te protège de tout mal.

A la famille du Club Espoir FSTF

Merci pour m'avoir toujours supporté. Merci pour tout votre amour et votre confiance, je vous dédie ce travail comme preuve de respect, de gratitude et de reconnaissance.

A ma famille, à mes amis, à tous ceux qui, par un mot, m'ont donné la force de continuer

Remerciement

Après Dieu le tout puissant et miséricordieux, je tiens à toutes les personnes, dont l'intervention au cours de ce projet, a favorisé son aboutissement.

Je tiens à remercier toute personne qui a participé de près ou de loin à la réussite de ce modeste travail et à la tête de ces personnes mes chers encadrants :

Mme CHAKER Ilham

Avec mon admiration pour votre rigueur dans le travail, pour vos qualités humaines et professionnelles. Je vous prie de trouver dans ce travail toute la reconnaissance que je vous témoigne.

Mr ZARGHILI Arsalane

Pour son soutien, son aide, et ses conseils qui m'ont guidé durant l'élaboration de ce travail, je vous prie de trouver dans ce travail un hommage vivant à votre haute intelligence, sagesse et personnalité.

Mes vifs remerciements vont également aux membres du jury, **Pr KHARROUBI Jamal, Pr BENABBOU Rachid, Pr ABBAD Khalid**, qui ont accepté d'examiner ce travail et de l'enrichir par leurs propositions.

Pour ton aide et soutien. Je vous prie, **Mr LASRI Younes** - Doctorant au laboratoire SIA - de trouver dans ce travail toute la reconnaissance que je vous témoigne.

À tous mes enseignants, j'ai su apprécier la qualité de l'enseignement que vous m'avez transmis, vos compétences et vos rigueurs scientifiques sont pour moi une référence. Je vous prie de trouver ici l'expression de mes vifs remerciements.

Merci à tous ceux qui ont participé de près ou de loin pour la réalisation de ce travail.

Sommaire

Résumé

La segmentation des caractères arabes est une étape nécessaire dans la reconnaissance optique de caractères optique (OCR). La nature cursive du texte arabe pose des problèmes difficiles dans la reconnaissance du caractère arabe ; Cependant, les caractères incorrectement segmentés entraîneront des erreurs de classification des caractères qui, à leur tour, peuvent conduire à de mauvais résultats de reconnaissance. Par conséquent, la segmentation des caractères arabes hors ligne reste un problème de recherche vaste et peu de recherches ont été réalisées dans ce domaine au cours des dernières décennies. Ceci est dû à la fois à la nature cursive de l'écriture de l'Arabe dans les formes imprimées et manuscrites et le manque de bases de données et de dictionnaires arabes. La plupart des méthodes utilisées dans la reconnaissance des caractères arabes sont adoptées à partir des méthodes disponibles utilisées pour des caractères latins et chinois manuscrits ; Cependant, d'autres méthodes sont développées uniquement pour la segmentation des caractères arabes.

Le présent travail décrit en premier temps les différentes caractéristiques de l'alphabet arabe. Il donne également une brève description des travaux déjà réalisés dans le domaine de segmentation de texte arabe imprimé, en les classifiant selon leurs approches de base. Ce rapport fournie également une étude détaillée et comparaison de quelques méthodes de segmentation. Enfin un nouveau processus de segmentation du texte arabe en caractères individuels est proposé. Ce système de segmentation hors ligne est réalisé de façon à segmenter les textes multi-fonte et multi-taille.

A cause de l'absence d'un corpus de référence pour les systèmes de segmentation, nous présentons également dans ce rapport une description d'un corpus unifié ; que nous proposons pour être utilisé par tous les systèmes de segmentation du texte arabe imprimé.

Mots clés : Segmentation, texte arabe imprimé, caractère, corpus, contour, template-matching, projection, python, OpenCV.

Abstract

Arabic character segmentation is a necessary step in Arabic Optical Character Recognition (OCR). The cursive nature of Arabic script poses challenging problems in Arabic character recognition; however, incorrectly segmented characters will cause misclassifications of characters, which in turn may lead to wrong results. Therefore, off-line Arabic character segmentation is a difficult research problem, and little research has been achieved in this area in the past few decades. This is due to both the cursive nature of Arabic writing in both printed and handwritten forms and the scarcity of Arabic databases and dictionaries. Most of the character recognition methods used in the recognition of Arabic characters, are adopted from available methods used on handwritten Latin and Chinese characters. However, other methods are developed only for Arabic character segmentation.

This work describes, firstly, the different characteristics of the Arabic alphabet. It gives also a description of the works carried out in Arabic text segmentation, by classifying them according to their basic approaches. This report manuscript provides also a detailed study and comparison of some segmentation methods. Finally, a new process of segmentation of the text is proposed. This offline segmentation system is designed to segment multi-fonte and multi-size Arabic texts into individual characters.

Due to the lack of reference corpus for all segmentation systems, we present in this work a description of our proposed reference corpus. This corpus can be used by all segmentation systems of printed Arabic text.

Key words: Recognition, Segmentation, Arabic text, character, corpus, contour, Template-matching, projection, python, OpenCV.

ملخص

تعتبر تجزئة النص العربي إلى حروف منفصلة مرحلة أساسية من مراحل برامج التعرف على النصوص. فالطبيعة المترابطة للحروف العربية تفرض مشاكل كبيرة لبرنامج تجزئة الحروف العربية. وبالتالي تنتج أخطاء عديدة أثناء التعرف عليها. مما يسبب يؤثر سلبا في المراحل المتقدمة من منظومة إعادة التعرف على النص العربي. لهذا تعتبر تجزئة الحروف العربية مجالا خصبا للأبحاث العلمية مع أن القليل من الأبحاث تمت في هذا الاتجاه خلال العقدين الأخيرين. السبب في هذا يرجع لكون الحروف العربية تكتب بطريقة مترابطة في حالة النصوص المطبوعة آلياً كما في حالة النصوص المكتوبة يدوياً. سبب أخر لقلة الأبحاث في هذا المجال يرجع لضعف وقلة المراجع من المعاجم اللغوية وقواعد البيانات.

معظم أساليب التعرف المستخدمة في التعرف على الحروف العربية، يتم اعتمادها من الأساليب المتاحة للتعرف على الحروف اللاتينية أو الصينية. مع ذلك فهناك من الأساليب ما تم تطويره فقط للتعرف على الحروف العربية. في هذا المجال أجرينا بحث التخرج هذا، والذي يقدم أولا، وصفا شاملا لخصائص الحروف الأبجدية العربية. كما يقدم تحليلا بسيطا لبعض الأبحاث المنجزة لتقسيم النصوص العربية. هذه الأبحاث تم تصنيفها حسب الأساليب المعتمدة خلال عملية التجزئة. ثم يقدم هذا التقرير دراسة مفصلة لبعض الأبحاث المختارة من خلال الوقوف على أهم المشاكل التي قد تواجهها في عملية التجزئة. أخيرا وليس آخرا، نقترح بالتفصيل، طريقة جديدة لتقسيم النص العربي إلى حروف. الطريقة المقترحة في بحث التخرج هذا، موجهة للتعامل مع النصوص العربية المكتوبة بمختلف الخطوط وبمختلف الأحجام. وفي أخر فقرات هذا التقرير نناقش أهم النتائج التي تحصلنا عليها خلال هذا البحث، ونقف على الحالات الشاذة التي لم نتمكن من معالجتها.

كما أن عدم توفر قاعدة بيانات موحدة لأنظمة تجزئة النص العربي، دفعنا لإنشاء قاعدة بيانات مرجعية يقدم هذ التقرير وصفا لها. قاعدة البيانات هاته يمكن ان تستعمل في جميع أنظمة التي تجزئ النص العربي المخطوط آليا إلى حروف منفصلة.

الكلمات الدلالية: تجزئة، النص العربي، الحروف العربية، التعرف على الحروف . قاعدة بيانات.

Liste des figures

Liste des tableaux

Liste des acronymes

OCR	Optical Character Recognition
RGB	Red Green Blue
SIA	Systèmes Intelligents et Applications
FST	Faculté des Sciences et Techniques
AOCR	analytic Optical Character Recognition
Dpi	dots per inch
OpenCV	Open Computer Vision
NumPy	Numeric Python
SciPy	Scientific Python
TCL	Tool Command Language
BSD	Berkeley Software Distribution
PIL	Python Imaging Library
Ppp	Points par pouce

Introduction générale

La reconnaissance optique de caractères, habituellement abrégée en OCR, a fait l'objet de recherches intensives depuis le développement des ordinateurs numériques. L'OCR implique des systèmes informatiques conçus pour traduire des images de texte en texte compréhensible et éditable par machine pour être utilisé dans de nombreuses applications telles que le traitement des documents, le traitement des chèques bancaires, le tri et le routage automatiques du courrier, la vérification des signatures…

OCR est un domaine de recherche en intelligence artificielle, reconnaissance de formes et vision par ordinateur. Au cours du siècle dernier, la recherche a surtout porté sur les caractères latins, chinois et japonais, et ce n'est qu'au cours des deux dernières décennies que d'autres langues ont été sérieusement étudiées. On parle de l'arabe, le farsi, Sindhi, Uyghur, etc.

Près de 500 millions de la population mondiale utilise la langue arabe en parlant, en écrivant et / ou en lisant, en plus de cela, presque tous les musulmans peuvent lire l'écriture arabe car c'est la langue d'Al-Quran, pourtant peu de progrès a été accomplis dans la recherche pour la reconnaissance automatique des caractères arabes. Par conséquent, beaucoup plus d'attention est nécessaire pour obtenir de meilleurs résultats. Cela est principalement dû au manque de financement de la recherche, de bases de données et de dictionnaires. D'autres complexités qu'un système de reconnaissance hors ligne doit traiter sont la mauvaise résolution du document, qui peut contribuer à la lisibilité lorsque les caractéristiques essentielles des caractères sont perdues. Reconnaître l'écriture arabe présente deux défis supplémentaires : L'orthographe est cursive et la forme de la lettre est sensible au contexte.

Les systèmes OCR varient dans la façon d'acquérir leurs entrées (en ligne/hors ligne), le mode d'écriture (manuscrit/imprimé), la restriction sur les polices (mono/multi fonte) et la connectivité du texte (caractères isolés ou mots cursifs).

Dans un système OCR typique pour le script cursif, les caractères d'entrée sont lus et numérisés par un scanner optique, et chaque caractère est alors localisé et segmenté, et la matrice résultante est introduite dans un préprocesseur pour le lissage, la réduction du bruit et la normalisation de la taille. Cette approche est connue comme une **approche analytique**, dans laquelle le mot est segmenté en unités classifiables plus petites (caractères). Il est clair que cette étape est difficile, en particulier dans le cas du script manuscrit. La réussite d'un système OCR arabe à reconnaitre les caractères dépend fortement de la qualité de la segmentation du texte en caractères individuels. Les erreurs de segmentation vont surement impliquer une fausse reconnaissance où le rejet de caractères dans les meilleurs des cas, ce qui ne peut pas être supporté dans un système critique tels que les applications de gestion des courriers…

Pour éviter les difficultés de la segmentation, les chercheurs ont découvert une autre approche, connue sous le nom d'**approche holistique** (ou globale), dans laquelle la reconnaissance est réalisée globalement sur la représentation complète des mots sans avoir recours à la segmentation, c'est pourquoi Il est également dite approche free-segmentation.

Dans ce travail nous présentons les caractéristiques du texte arabe, ensuite un bref aperçu sur la segmentation des textes arabes, les étapes et les meilleures méthodes de segmentation. La plupart de ces méthodes sont limitées à la segmentation du texte mono-fonte et mono-taille, c'est pour cela que nous introduisons ensuite notre contribution à la segmentation du texte arabe en caractères imprimés en **multi-fonte** et **multi-taille.** Notre contribution principale est sous forme d'une méthode hybride et robuste, qui se serve de l'analyse du contour et du balayage par fenêtre glissante, pour isoler les caractères du texte cursif. Nous présentons également dans ce rapport une description d'un corpus de référence, que nous avons proposé pour être utilisé par tous les systèmes de segmentation du texte arabe imprimé.

Dans le cadre de ce projet de fin d'étude, nous avons eu l'occasion de découvrir le domaine de recherche scientifique au sein de laboratoire LSIA (**L**aboratoire **S**ystèmes **I**ntelligents & **A**pplications) de la faculté des sciences et techniques de Fès, sous la direction de Pr ZARGHIILI Arsalane. Ce laboratoire accompagne les étudiants PhD et les étudiants de master tout au long de leur formation, en mettant à leur disposition tous les moyens nécessaires pour la réalisation de leurs projets.

Le laboratoire SIA, crée en 2011, est une unité de Recherche du Centre d'Etudes Doctorales en Sciences et Techniques de l'Ingénieur domicilié à la Faculté des Sciences et Techniques de Fès et regroupant des laboratoires de recherche tous accrédités par l'Université Sidi Mohamed Ben Abdellah de Fès, et domiciliés à la Facultés des Sciences et Techniques, l'Ecole Supérieure de Technologie, la Faculté Polydisciplinaire de Taza, l'ENS de Fès et la Faculté de médecine et pharmacie de Fès.

Organisation du rapport :

Ce rapport est constitué de quatre chapitres organisés comme suit :

- Le premier chapitre résume les différentes caractéristiques du texte arabe dans une première partie. Ensuite, nous présentons l'état de l'art de la segmentation du texte arabe en lignes, en pseudo-mots, et plus particulièrement en caractères.
- Le deuxième chapitre est une étude de quelques méthodes de segmentation, où nous discutons quatre méthodes que nous avons implémentées.
- Dans le troisième chapitre, nous présentons notre contribution à la segmentation des textes arabes **multi-fonte** et **multi-taille**, en proposant des améliorations pour les méthodes étudiées dans le deuxième chapitre.
- Finalement nous présentons la partie pratique de ce travail dans le dernier chapitre, et la discussion des résultats obtenus.

CHAPITRE 1. Etat de l'art sur la segmentation du texte arabe imprimé

Introduction

La segmentation est le processus d'isolement des caractères individuels à passer pour le système de reconnaissance. La segmentation du texte en caractères est l'étape la plus cruciale et la plus difficile pour un système OCR du texte arabe en fait, un processus de segmentation faible produit la mauvaise reconnaissance ou le rejet du caractère. Le processus de segmentation pour le problème de reconnaissance de caractères peut être divisé en trois niveaux : **segmentation en lignes, segmentation en mots ou pseudo-mots et segmentation en caractères**. La figure 1 illustre le processus de segmentation d'un système OCR.

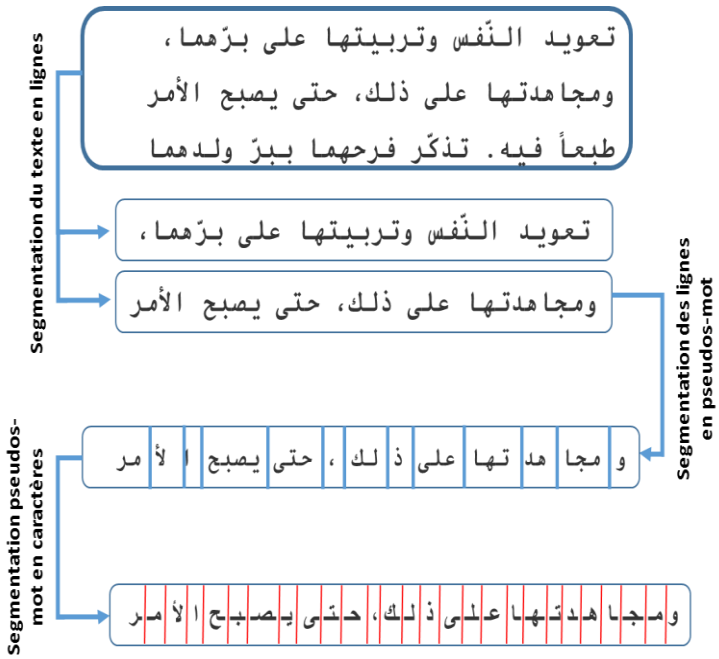


Figure 1: Le processus de segmentation

Plusieurs efforts ont été consacrés à la segmentation des textes cursifs, mais jusqu'à présent il s'agit toujours d'un problème non résolu, quel que soit l'algorithme utilisé, il doit traiter le problème de la cursivité et les effets des autres caractéristiques qui influent la segmentation.

1.1. Caractéristiques du texte arabe

1. Le texte arabe s'écrit de droite à gauche. Il contient 28 caractères, la forme de chaque caractère varie selon sa position dans le mot ; Chaque caractère peut avoir jusqu'à quatre formes différentes. Évidemment, cela va augmenter le nombre de classes à reconnaître de 28 à 121. Tableau.1 montre les 121 formes possibles des caractères arabes. Il faut noter que Les mots sont séparés par des espaces. De toute évidence, si les six caractères ا د و ز رذ, apparaissent au milieu du mot, celui-ci sera divisé en blocs de composants connectés appelés pseudo-mots.

Ainsi, un mot peut avoir un ou plusieurs pseudo-mots. Les pseudo-mots sont également séparés par des espaces, mais ces derniers sont généralement plus courts que ceux entre les mots : Figure 2 .Donc, ce problème doit être envisagé pour éviter de segmenter un mot en deux mots plutôt que pseudo-mots.

nom	isolé	début	milieu	fin	nom	isolé	début	milieu	fin
Alif	ا			ـا	Fa	ف	فـ	ـفـ	ـف
Ba	ب	بـ	ـبـ	ـب	Qua	ق	قـ	ـقـ	ـق
Ta	ت	تـ	ـتـ	ـت	Kaf	ك	كـ	ـكـ	ـك
Tha	ث	ثـ	ـثـ	ـث	Lam	ل	لـ	ـلـ	ـل
Jeem	ج	جـ	ـجـ	ـج	Mim	م	مـ	ـمـ	ـم
Hha	ح	حـ	ـحـ	ـح	Noun	ن	نـ	ـنـ	ـن
Kha	خ	خـ	ـخـ	ـخ	Haa	ه	هـ	ـهـ	ـه
Dal	د			ـد	Waw	و			ـو
Thal	ذ			ـذ	Yaa	ي	يـ	ـيـ	ـي
Ra	ر			ـر	Alif	آ			ـا
Zay	ز			ـز	Alif	أ			ـأ
Seen	س	سـ	ـسـ	ـس	Alif	إ			ـإ
Sheen	ش	شـ	ـشـ	ـش	Waw	ؤ			ـؤ
Sad	ص	صـ	ـصـ	ـص	Ya	ئ	ئـ	ـئـ	ـئ
Dhad	ض	ضـ	ـضـ	ـض	LamAlif	لا			ـلا
Ta	ط	طـ	ـطـ	ـط	LamAlif	لأ			ـلأ
Dza	ظ	ظـ	ـظـ	ـظ	LamAlif	لإ			ـلإ
Ain	ع	عـ	ـعـ	ـع	LamAlif	لآ			ـلآ
Ghain	غ	غـ	ـغـ	ـغ	Hamza	ء			

Tableau 1 : Toutes les formes possibles des caractères arabes

(a) عبد الحي (b) ا ز و ا ز و

Figure 2 : mot composé de 3 pseudo-mots (b) mot composé de caractères isolés

2. La longueur des caractères arabes est variable, par exemple (ك) et (ا) ; Ils diffèrent également en hauteur, par exemple : (د) et (ل). En outre, la largeur et la hauteur varient selon les différentes formes du même caractère dans différentes positions dans le mot, par exemple (ح) et (حـ). Cela veut dire que la segmentation basée sur une taille fixe n'est pas applicable.

3. Pour quelques polices, certaines combinaisons de lettres peuvent donner lieu à un nouveau symbole (ligature) : Figure 3. Ces formes et combinaisons multiples de lettres augmentent de manière significative le nombre de différents symboles représentés qu'un classifieur doit reconnaître à bien plus d'une centaine, en plus des signes de ponctuation et des chiffres.

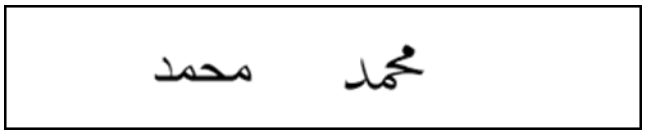


Figure 3 : (a) un mot sans ligature (b) le même mot avec ligature

4. Beaucoup de lettres arabes sont distinguées les unes des autres seulement par des points situant au-dessous ou au-dessus de la lettre Figure 4.a. Un mal classement peut conduire à un mot complètement différent.

5. Les signes diacritiques représentant les voyelles (ou courtes-voyelles) peuvent se trouver au-dessous ou au-dessus du mot. Les diacritiques –lorsqu'ils sont utilisés- déterminent la prononciation du caractère et la signification du mot, par exemple le mot « كتب » peut signifier « écrire – كَتَبَ» ou « livres - كُتُبٌ » ou « a été écrit – كُتِبَ » et d'autres significations. Mais ces signes sont souvent laissés de côté et la signification d'un mot est identifiée à partir du contexte, Figure 4.b résume tous les signes de voyellation possibles.

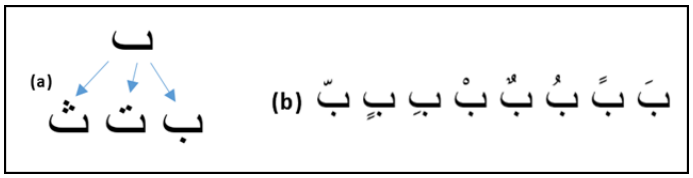


Figure 4 : (a) exemple de caractères avec les points (b) : une même lettre avec des signes diacritiques différents

Les caractères arabes sont connectés par une ligne de base, et l'espace est utilisé comme séparateur de mots. La ligne de base est horizontale et traverse des parties connectées du texte. La ligne de base présente le nombre maximal des pixels de texte ; la figure 5 montre la ligne de base d'une ligne de texte arabe. La connaissance de l'emplacement exact de la ligne de base améliore la capacité du système OCR de séparer le texte en lignes.

Figure 5 : la ligne de base d'un texte arabe

1.2. Segmentation en lignes

Le document contenant le texte est scanné dans un premier temps puis converti en niveau de gris. Après cette phase d'acquisition, le système effectue plusieurs opérations de prétraitement pour préparer le document à la segmentation et puis la reconnaissance. Ces opérations peuvent

être : élimination des niveaux RGB, ajustement de l'éclairage, élimination du bruit, normalisation de la taille. Puis le système localise le texte dans l'image et effectue la segmentation.

Pour segmenter le texte en lignes, presque tous les systèmes AOCR (Arabic Optical Character Recognition) font appel à la méthode de la Projection Horizontale [35], qui calcule l'histogramme horizontal de l'mage, et si le nombre de lignes blanches successives rencontrées –du haut vers le bas- est supérieur à un seuil à fixer, le nombre de lignes est augmenté, puis le système extrait le début et la fin de chaque ligne. La Figure 6 montre un exemple de la méthode utilisée.

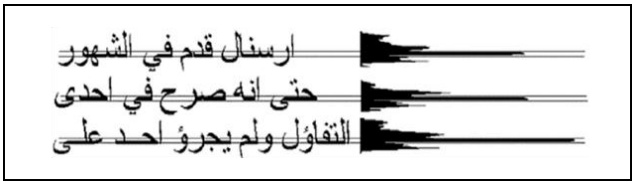


Figure 6 : Exemple de la segmentation du texte en lignes par projection horizontale

1.3. Segmentation en pseudo-mots

La ligne de texte est segmentée en mots ou en pseudo-mots. Ceci est fait en utilisant la projection verticale de l'image [35]. Cette dernière est simple et similaire à la projection horizontale. La seule différence entre les projections horizontales et verticales est que dans ces dernières, nous comptons le nombre de pixels noirs sur chaque colonne de l'image texte. Si le nombre de pixels noirs n'est pas égal à zéro, il indique une partie connectée et par conséquent le texte n'est pas segmenté. D'autre part, si le nombre de colonnes blanches successives rencontré est supérieur à un seuil, la partie courante est segmentée. La figure 7 montre le profil de la projection verticale pour une image donnée.

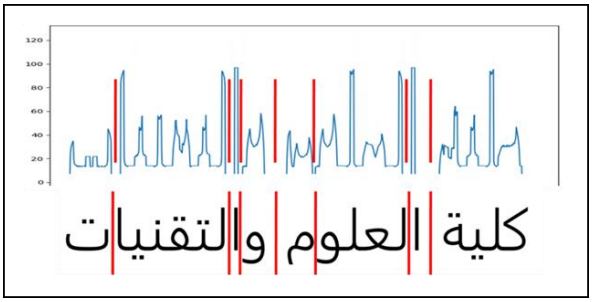


Figure 7 : Exemple de la segmentation d'une ligne de texte en pseudo-mots par projection verticale

1.4. Segmentation en caractères

1.4.1. Généralités

L'étape de segmentation des mots en caractères est une étape cruciale dans les systèmes OCR analytiques. Une erreur dans la segmentation des caractères produira des erreurs dans la reconnaissance.

Un point de segmentation est un point entre deux caractères, il se trouve sur la ligne de base qui ne contient aucune information. La tâche de segmentation consiste à localiser deux points de segmentations, puis extraire le caractère entre eux.
Dans cette partie nous allons présenter des techniques de segmentation des mots arabes imprimés en caractères.

Les travaux de Parhami et Taraghi [BM82] en 1981 suivis par l'œuvre d'Amin et Masini [AM82] en 1982 ont été les premières tentatives de segmenter les caractères arabes. Dans les systèmes OCR analytiques, plusieurs techniques ont été proposées pour la segmentation des mots en caractères ; dans les premières tentatives, la projection verticale a été utilisée à cet effet. Plus tard, la tendance était d'obtenir le squelette du mot et de l'analyser pour trouver les points de segmentation appropriés. Cette méthode a été suivie de tentatives de segmentation des mots en traçant le contour du mot. Plusieurs d'autres recherches se sont basées sur la technique de Template-Matching seule ou combinée avec les autres techniques citées précédemment. Les systèmes OCR globaux utilisent l'approche holistique ou sans segmentation dans laquelle les mots sont reconnus sans segmentation. Le schéma de la Figure 8 montre notre classification des techniques utilisées dans le domaine de la segmentation du texte arabe.

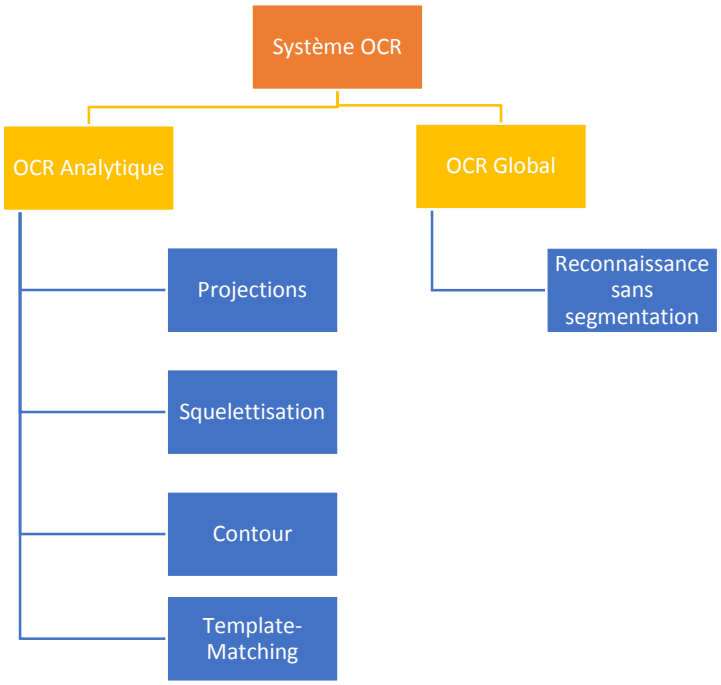


Figure 8 : classification des techniques utilisées dans le domaine de la segmentation du texte arabe

Dans ce qui suit, des travaux de segmentation sont regroupés selon la technique utilisée.

1.4.2. Segmentation basée sur les projections

Le but de la méthode de projection est de simplifier un système OCR en réduisant les informations 2D en 1D. elle fonctionne mieux avec les documents imprimés, en particulier avec les polices qui ne forment pas des ligatures telles que «arabe transparent» et «arabe simplifié», cependant, pour les polices comme «arabe traditionnel» qui contient de nombreuses formes de ligature, la méthode des projection présente des problèmes inévitables.

Ces méthodes sont basées sur le fait que les traits de connexion sont toujours de moins d'épaisseur que d'autres parties des mots. Dans ces procédés, on calcule la projection verticale de l'image :

$$Vi=\sum P(i,j)$$

Où p(i,j) est la valeur de pixel à la position (i,j) qui est soit 0 (blanc) soit 1 (noir).

L'ensemble V des Vi forme l'histogramme de l'image, qui peut être manipulé de plusieurs manières pour déterminer les points de segmentation.

Zheng et al. [AHT04] ont proposé un nouvel algorithme de segmentation des caractères arabes, qui est basé sur l'histogramme vertical et quelques autres règles, De plus, les caractéristiques structurelles entre les régions de fond et les composants de caractère, les caractéristiques des caractères arabes, sont également utilisées pour vérifier si le pseudo-mot ne comprend qu'un seul caractère. Ensuite, l'histogramme vertical, et d'autres règles ont été utilisés pour trouver des points de segmentation réels. Enfin, les pseudo-mots ont été divisés en points de segmentation. Les résultats expérimentaux montrent que l'algorithme atteint environ 94% de segmentation correcte.

Dans [NSZA03] et [NSA03] les chercheurs ont utilisé la projection verticale de la zone centrale au lieu du mot entier. Quatre zones de ligne de texte ont été identifiées ; À savoir la ligne de base, la zone centrale, et la zone inférieure. Si la valeur de la projection verticale de la zone centrale est inférieure à deux tiers de l'épaisseur de ligne de base, la zone est considérée comme une zone de connexion entre deux caractères. Alors que toute zone suit la zone de connexion avec une plus grande valeur étant considérée comme le point de départ d'un nouveau caractère tant que le profil est supérieur à un tiers de la ligne de base. La Figure 9 montre les étapes de cette méthode.

Cette technique a été utilisée avec la police Naskh et la sur-segmentation a été observée avec certains caractères tels que (ش), (س). Le problème a ensuite été résolu dans la phase de reconnaissance en combinant plus d'un segment pour produire le caractère correct.

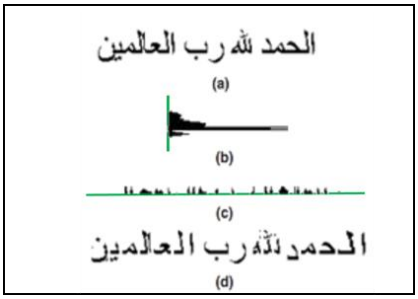


Figure 9 : Processus de segmentation (a) Texte original en arabe (b) Profil de projection horizontale (c) Profil de projection verticale de la zone intermédiaire (d) Caractères segmentés

Amin et Mari [AM89] ont utilisé la valeur moyenne au lieu de la simple projection verticale. Le point de connectivité montre la valeur la plus faible de la valeur moyenne (AV), où :

$$AV = \frac{\sum_{i=1}^{Nc} Xi}{Nc}$$

Où N_c est le nombre de colonnes et X_i est le nombre de pixels noirs de la $i^{ème}$ colonne. Par conséquent, chaque partie présentant une valeur de somme inférieure à AV doit être segmentée en un caractère différent Figure 10. La même méthode a été adoptée dans [I03] et [AS91]. Cependant, comme la formule ci-dessus a surestimé le nombre de points de connexion, deux autres règles ont été incluses pour éliminer certains points estimés incorrectement, c'est-à-dire si la projection verticale ne suit pas la règle suivante, le caractère n'est pas segmenté :

$$|d_i| < d_L /3$$

Où d_i est la distance entre le $i^{ème}$ pic et le $i+ 1^{eme}$ pic, et d_L est la largeur totale du caractère. En examinant les caractères arabes, la distance entre les pics ne dépasse pas 1/3 de la largeur du caractère arabe.

De plus, à la fin d'un mot ou pseudo-mot, la règle suivante est appliquée :

$$L_{i+1} > 1.5*L_i$$

Où L_i est le $i^{ème}$ pic dans l'histogramme. Cette règle est mise en œuvre en raison de l'inter-connectivité des caractères arabes et de leurs formes à la fin d'un mot.

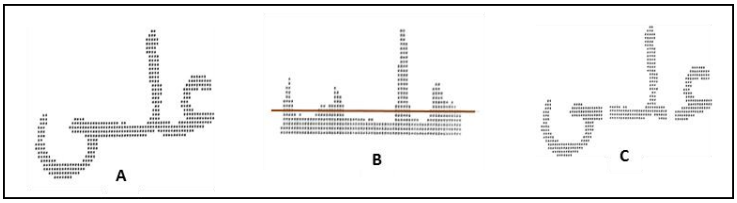


Figure 10 : Exemple de mot arabe et sa segmentation en caractères. (A) mot arabe. (B) projection. (C) Word segmenté en caractères.

Dans [L14], L. Younes a appliqué une projection verticale simple sur le mot après avoir éliminé la ligne de base, cette méthode a rencontré plusieurs difficultés telles que la détection de la ligne de base comme étant la partie qui présente le nombre maximal des points noirs dans la direction horizontale, ainsi, la difficulté de savoir si le mot contient une ligne de base, ou s'il est constitué seulement des caractères isolés et le traitement nécessaire dans ce cas. Figure 11 montre le processus suivit.

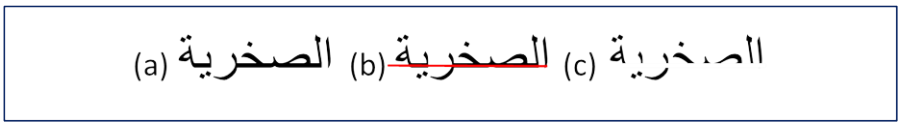


Figure 11 : Processus de segmentation (a) mot original (b) détection de la ligne de base (c) élimination de la ligne de base

Il est clair que cette technique d'analyse des projections dépend fortement d'une valeur de seuil prédéfinie liée à la largeur des caractères, et à la fonte utilisée. Il faut noter également que les éléments secondaires tels que les points et les signes diacritiques, influent sur la segmentation. Et par la suite, il faut les éliminer pour avoir une projection du caractère brut.

On conclut que les méthodes de segmentation qui utilisent l'histogramme de projection verticale dépendent fortement de la détermination de la ligne de base. Ces méthodes sont indépendantes de la forme, la taille ou de la police des caractères lorsque cette dernière ne contient aucun chevauchement. Elles sont les mieux adaptées pour les caractères imprimés, tout en étant insuffisants pour segmenter les caractères superposés ou un script manuscrit car les ces types de texte contiennent fréquemment des variations de la ligne de base. En outre, cette approche ne fonctionnera pas efficacement pour les images déviées et tous les pseudo-mots en chevauchement ne peuvent pas être séparés par cette méthode.

1.4.3. Segmentation basée sur la squelettisation

Les informations essentielles d'une forme sont stockées dans son squelette. De nombreux algorithmes ont été proposés pour extraire le squelette, tels que la transformée de distance, l'amincissement homotypique avec ses deux approches : séquentielle et parallèle comme l'algorithme de Zhang et Suen [TC84]**.**

La squelettisation est une technique essentielle qui pourrait aider à résoudre le problème de segmentation de caractères ; Cependant, la combinaison avec d'autres techniques serait essentielle pour garantir une meilleure segmentation. La figure 12 montre un exemple d'un mot arabe et son squelette.

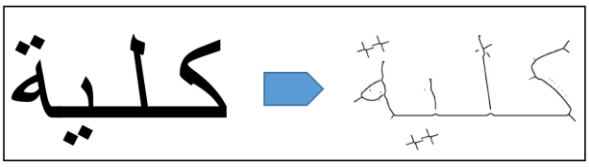


Figure 12 : squelettisation

Dans la méthode d'El-Khaly et Sid-Ahmed [ES90], la ligne de base du mot aminci (squelette) est trouvée en première étape, c'est la ligne qui contient le nombre maximal des points noirs. Puis seulement les colonnes qui n'ont pas de pixels au-dessus ou en dessous de la ligne de base sont considérés pour trouver les points de segmentation. Le point de segmentation sera au milieu du segment de connexion.

La procédure de segmentation présentée dans [1GUA92] décrite par H.Goraine, M. Usher, est similaire à celle d'AIEmami dans [S88]. L'idée de l'algorithme, après avoir obtenu le squelette du mot en utilisant l'algorithme de Hilditch [01], est de trouver le point de départ en utilisant l'organigramme de la figure 13.a tout en se basant sur une classification des points rencontrés : point de connexion, point caractéristique ou jonction, trait. La deuxième étape est de trouver le point final et ensuite de tracer un trait depuis le début jusqu'à la fin. Figure 13.b. Par la suite, le tracé est isolé et éliminé de l'image. Le premier point (à droite) détecté est pris comme point final de du tracé suivant, et la procédure de segmentation est répétée jusqu'à ce qu'il n'y ait plus de traits.

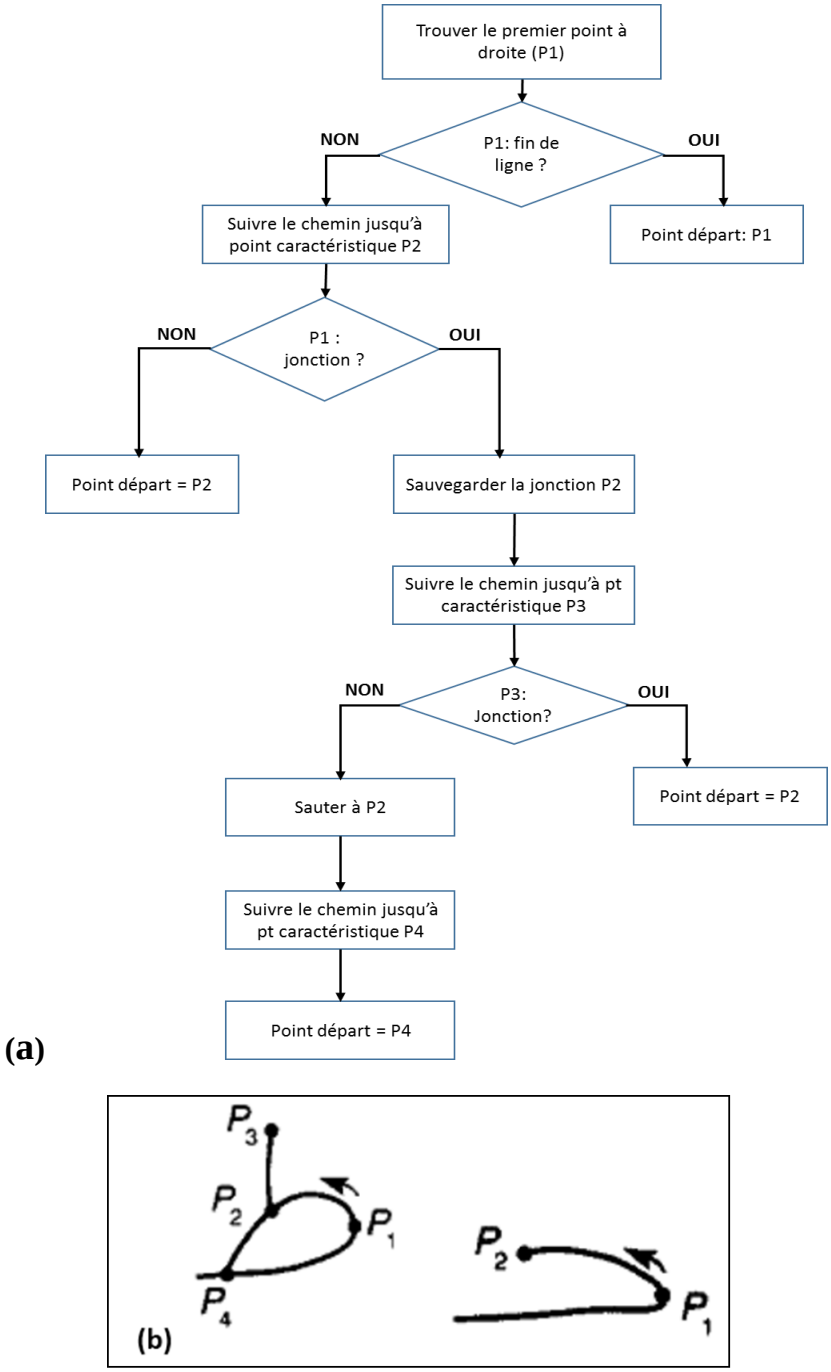


Figure 13 :(a) Flow-chart de détermination du point de départ. (b) construction du caractère

Al-Sadoun et Amin [HA95] ont tracé le squelette de droite à gauche en utilisant une fenêtre 3*3 pour identifier les points potentiels de segmentation. Ensuite, un arbre binaire est construit et le squelette est représenté à l'aide du code de Freeman [H68]. Chaque nœud de l'arbre binaire décrit la forme de la partie correspondante du pseudo-mot. L'arbre binaire est lissé pour minimiser le nombre de nœuds en éliminant les nœuds vides, en minimisant la chaîne de code Freeman et en éliminant (en minimisant) tout bruit dans l'image amincie.

Enfin, l'arbre binaire est segmenté en sous-arbres de sorte que chaque sous-arbre décrit un caractère en utilisant des primitives, y compris des lignes, des boucles et des boucles doubles. Certaines règles ont été définies pour assurer les limites correctes de caractères tels que : long

segment horizontal signale la fin du caractère actuel et l'existence de boucles ou un long segment vertical sont considérés comme le début d'un caractère. L'algorithme peut être appliqué à n'importe quelle police et la taille du texte arabe, en plus, il peut être appliqué au texte manuscrit. Un avantage de cette méthode est que l'identification de la ligne de base devient inutile puisque le pseudo-mot est décrit par un arbre binaire, ce qui permet d'économiser du temps de traitement.

Parmi les inconvénients des méthodes basées sur la squelettisation, les différents algorithmes d'amincissement peuvent produire différents caractères amincis, de plus, le processus d'amincissement peut modifier la forme du caractère, en particulier dans le cas des images de mauvaise qualité qui, à leur tour, rendent le caractère difficile à reconnaître. Certains problèmes courants rencontrés pendant le processus de squelettisation comprennent l'élimination ou l'érosion des caractères secondaires. Ces modifications rendent la reconnaissance de l'image du squelette une tâche difficile même pour le traitement visuel humain de la nature. Plus de problèmes dans [JF01].

1.4.4. Segmentation basée sur le contour

La détection des contours d'une image réduit de manière significative la quantité de données et élimine les informations qu'on peut juger moins pertinentes, tout en préservant les propriétés structurelles importantes de l'image. Cette approche a beaucoup d'avantages par rapport à l'utilisation du squelette du mot dans lequel l'information de mot peut être perdue, et qui conduit à moins de taux de reconnaissance.

Dans [BS97] Le contour supérieur est examiné pour les points candidats pour la segmentation des pseudo-mots. Cela se fait en traçant cette partie de gauche à droite en commençant par le premier point au-dessus de la ligne de base. Lorsqu'un point maximum dans la direction verticale est atteint, il est considéré comme un pic si sa valeur est supérieure à une valeur du seuil (t1 = ligne de base + t/6) avec t est la distance entre le top du mot et la ligne de base. Après cette étape, la valeur des coordonnées de contour commence à descendre jusqu'à ce qu'elle arrive à un point minimum : Figure 14. Si cette valeur est inférieure à la valeur de seuil (t1), elle est considérée comme un point de segmentation à condition qu'elle soit suivie d'un pic. Cette procédure se poursuit jusqu'à ce que toutes les coordonnées de la partie supérieure du contour soient examinées. Le point de segmentation est considéré entre deux pics. Si aucun pic n'a été trouvé après avoir rencontré un point minimum, ce point minimum est négligé.

De plus, si deux points minimum ou plus sont trouvés entre deux pics et les deux satisfont la condition de seuil, le point le plus proche du premier point de crête et le plus proche de la ligne de base est pris comme point de segmentation. Cette procédure se poursuit jusqu'à ce que tous les points de segmentation soient trouvés.

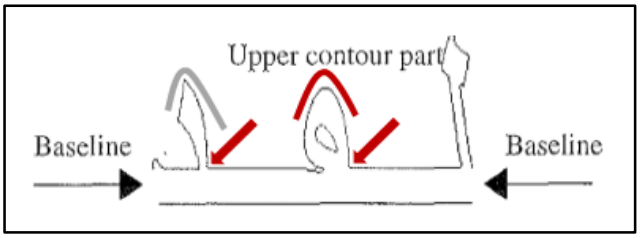


Figure 14 : exemple de localisation des points de segmentation (flèche)

La partie inférieure du contour est d'abord examinée pour voir s'il y a des caractères touchants ou s'il existe un (ى). Les caractères arabes peuvent se toucher sous la ligne de base, comme le montre la figure 15.a.

La segmentation des caractères touchants est obtenue en traçant la partie inférieure du contour de droite à gauche. Les valeurs les plus basses dans la direction verticale sont enregistrées. Le point de contact se trouve entre deux de ces points et est la valeur la plus élevée dans la direction verticale qui satisfait la condition de seuil mentionnée avant. Une fois qu'un point touchant est trouvé, les caractères sont séparés. Cela conduit à diviser le contour en deux parties ou plus en fonction du nombre de caractères touchants. Par conséquent, le contour de la première partie est extrait à nouveau. La partie inférieure de ce contour est examinée pour l'apparition du caractère (ى)

Dans le cas idéal, le contour du caractère (ى) rencontre à la ligne de base quatre fois comme le montre la Figure 15.b.

Cependant, cela peut varier en fonction de l'effet du bruit dans l'image. Lorsque cette condition n'est pas satisfaite, les changements de signe dans la direction horizontale sont considérés. Cinq changements de signe ou plus indiquent la présence de ce caractère. Lorsque ce caractère est identifié, les deux caractères sont séparés en un point entre eux comme le montre la Figure 15.b

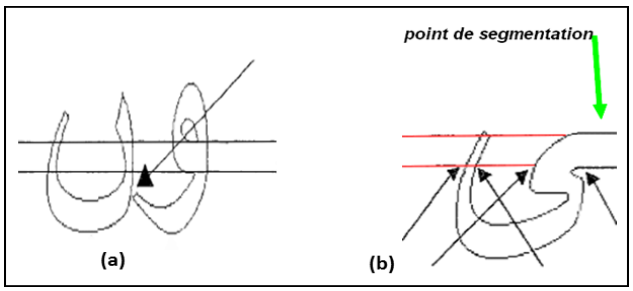


Figure 15 :(a) caractères touchants sous la ligne de base, (b) segmentation du caractère ى

La segmentation dans [M92] est basée sur le contour du corps principal des mots. Dans une première étape, le point de début et de fin du contour supérieur est déterminé. Il est important de trouver le point inférieur droit et inférieur gauche du contour en raison des longues lignes verticales au début d'un mot. Ensuite, une segmentation du contour supérieur en parties se fait par une courbure du même signe. En commençant par une courbure positive par exemple, le changement à une courbure négative va terminer ce segment et commencer par un nouveau. Un filtre passe-bas sur les points de contour sert à réduire la sensibilité au bruit de cette procédure. Dans certains cas, la ligne horizontale entre les caractères diffère beaucoup, mais cette longueur ne contient aucune information. Ensuite, un détecteur de ligne horizontale est

utilisé pour marquer ces lignes comme sans importance pour le processus de reconnaissance. La classification dans ce système requiert des caractères segmentés et cette segmentation se réalise à l'étape de la reconnaissance. Un caractère ne peut être segmenté que si la classification des segments correspondants est réussie. Le taux de segmentation n'a pas été mentionné, mais le taux de la reconnaissance était de 96.9%

Peng et al. Dans [LCXH06] ont proposé un schéma pour la segmentation des caractères imprimés en arabe sur la base de l'analyse des caractéristiques du contour de caractères. Tout d'abord, la position de la ligne base d'un mot donné est estimée, et une fonction D(x) de la distance entre le contour et la ligne de base est décrit : Figure 16. Les points de segmentation candidats sont ensuite détectés en analysant les caractéristiques de la courbe D(x). Les chercheurs n'ont pas mentionné le processus de détection des points de segmentation. Enfin, des règles structurelles sont proposées pour fusionner des caractères sur-segmentés. La conclusion de ce travail est qu'il est difficile de distinguer les caractères similaires trouvés dans différentes langues. La recherche n'a pas fourni de taux de reconnaissance spécifique sur le texte arabe, mais a déclaré des résultats concernant d'autres langues, Uyghur / Chinois / Sindhi... Les caractères Uyghur sont semblables à ceux de l'Arabe et le taux a été estimé à environ 98%.

Figure 16 : distance entre le contour et la ligne de base

C. Olivier, H. Miled dans [CHKY96] ont proposé une méthode qui utilise le contour supérieur pour segmenter le texte arabe manuscrit. Tout d'abord, ils ont détecté le contour de chaque pseudo-mot avec le code de Freeman [H68]. Le contour est étudié dans le sens inverse des aiguilles de la montre, ensuite le contour supérieur est déterminé par les directions F3. F1 et F5. Figure 17 Un filtre est appliqué pour ne garder que les points du contour –supérieur- situés dans la partie supérieure du pseudo-mot. Puis un automate est utilisé pour détecter les minimums locaux du contour supérieur, ces minimums sont pris pour la segmentation du pseudo-mot. Dans ce travail, C. Oiivier, H. Miled ont évalué le système sur une base de données contenant 6000 mots arabe, qui sont les noms du peuple tunisien écris par 20 fontes différentes. Ils ont pu avoir un taux d'erreur de 2.59%.

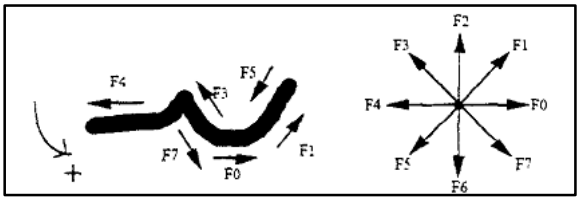


Figure 17: détection du contour supérieur en utilisant les codes de Freeman

La méthode présentée par Sari et al. [SS02] a utilisé le contour pour détecter les points de segmentation en appliquant des règles aux minimums locaux du contour inférieur de chaque pseudo-mot. Les caractères, verticalement chevauchés en raison de la nature de la fonte ou du style d'écriture (italique), ont été traités dans un stade avancé. Le taux de réussite était de 86% sur un ensemble de données limité de 100 mots seulement. Par conséquent, la méthode présentée ci-dessus montre la nécessité de combiner les fonctionnalités de contour avec d'autres fonctionnalités afin d'obtenir une meilleure segmentation des caractères, ce qui donnerait finalement un taux de reconnaissance plus élevé pour les systèmes OCR.

Le contour reste le meilleur choix à utiliser pour isoler les caractères connectés, et ceux en chevauchement. Les résultats de cette technique (contour) sont très remarquables par rapport aux autres techniques vus précédemment, mais la segmentation basée sur la détection du contour, a ses points faibles comme elle a ses points forts. Parmi les problèmes qu'on peut rencontrer lors de l'analyse du contour pour la segmentation, et la discontinuité de celui-ci, et cela peut être dû au bruit, c'est pour cela que la phase de prétraitement doit éliminer au maximum le bruit présent dans l'image. Un autre problème pour les systèmes qui utilisent en plus du contour, la ligne de base pour la segmentation, la détection de la ligne de base devient très difficile, et ses problèmes influent sur tout le système. Un autre problème de l'utilisation du contour pour la segmentation, c'est qu'il y a plusieurs méthode de détection de contour tels que le filtre de Sobel, filtre de Prewitt, filtre de Canny [02], ou l'annulation du Laplacien, et chaque méthode peut donner un contour diffèrent. Et il faut également noter que les systèmes de la segmentation basés sur le contour montrent une très faible performance face aux ligatures.

1.4.5. Segmentation basée sur Template-Matching

Template-Matching est une technique de traitement d'image numérique pour la recherche de petites parties d'une image qui correspondent à une image de modèle. Template-Matching a été utilisé dans les applications de reconnaissance de caractères. Elle est utilisée comme une technique pour reconnaître des caractères ou des mots avec une référence à une base de données stockée contenant un ensemble d'images pour les caractères ou les mots. Cette technique peut être considérée comme une technique de segmentation, mais la partie du caractère qu'on cherche dans l'image doit être choisie manuellement pour être utilisée à la comparaison.

Dans [BS97'], une technique a été proposée pour chercher l'apparition d'un angle formé par la jonction de deux caractères à la ligne de base Figure 18, en utilisant une fenêtre 7 × 7, pour examiner le voisinage des caractères. Cet angle est pris comme un point de segmentation potentiel. Pour valider les points précédemment calculés, le caractère extrait est comparé avec un model déjà stocké manuellement.

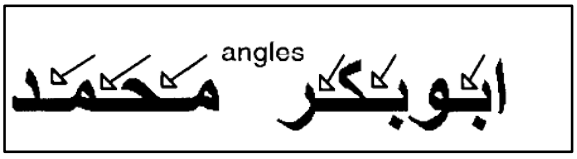


Figure 18: les angles de jonction entre les caractères

Dans [BRG14], Bharatratna P, Ramesh R et Ganesh M ont utilisé la technique du Template-Matching dans un système de segmentation automatique d'une scène vidéo pour séparer le script du reste de la capture. La Figure 19 montre le système utilisé. Celui-ci a été testé sur 35 vidéos avec 700 images pour chaque vidéo. Le résultat expérimental en termes de précision du système est de 91,52%.

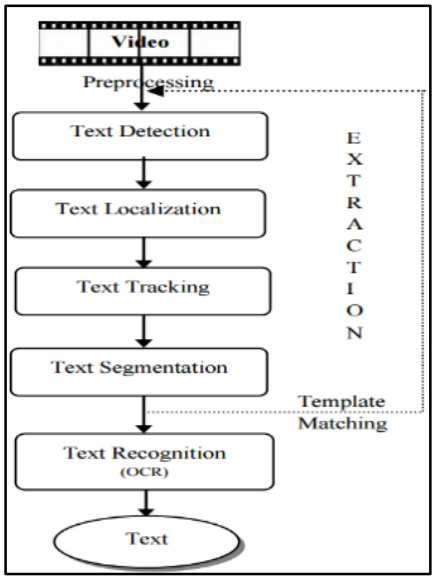


Figure 19 : système de segmentation du texte à partir d'une capture vidéo [BRG14]

Bien que la Template-Matching ait pu obtenir des bons résultats, sa réussite à trouver l'angle approprié dépend fortement du bruit dans l'image.

Template-Matching n'est pas une technique appropriée pour la segmentation du manuscrit et/ou imprimé arabe en raison de la diversité des fontes et les tailles d'écriture.

Conclusion

Dans ce chapitre nous avons vu les différentes caractéristiques de l'alphabet arabe, et les différents niveaux de segmentation du texte arabe. Et nous avons cité les quatre approches les plus utilisées pour la segmentation des mots arabes en caractères individuels, en donnant plusieurs exemples de travaux pour chaque approche. La majorité de ces travaux traitent la segmentation du texte écrit en une seule fonte.

CHAPITRE 2. Etude et implémentation de quelques méthodes de segmentation

Introduction

Après avoir analysé les différentes approches utilisées pour la segmentation du texte arabe en caractères dans le chapitre précèdent, dans ce chapitre nous avons choisi trois approches à implémenter et à étudier : Les projections verticales et horizontale, pour la segmentation du texte en lignes et la segmentation des lignes en pseudo-mots, et également pour la segmentation des pseudo-mots en caractères, l'approche du Template-Matching, et l'approche du contour. Nous discutons en détails les résultats et les problèmes qui peuvent être rencontrés pour chaque approche.

2.1. Segmentation du texte en lignes

2.1.1. Principe

La première étape dans le processus de segmentation consiste à séparer les lignes du texte présent dans l'image binaire acquise, et pour ce faire, nous avons implémenté la méthode de la **projection horizontale**. Nous avons calculé le nombre de pixels noirs dans chaque rangée de l'image, et nous avons construit un tableau d'une largeur égale à la hauteur de l'image. Ce tableau représente l'histogramme horizontal de l'image. La Figure 20 montre un exemple de la projection horizontale d'un texte de plusieurs lignes.

En analysant l'histogramme obtenu, le début d'une ligne de texte est détecté si le nombre des zéros successifs dépasse un seuil (trois) et est suivit d'une suite des non-zéros (Figure 21.a). De la même manière la fin d'une ligne de texte est détectée si au moins un non-zéro est suivi d'une suite d'au moins 3 zéros (Figure 21.b)

Après avoir localisé et sauvegardé les débuts et les fins des lignes de texte, nous extrayons l'image de chaque ligne sous forme d'une partie de la matrice de l'image binaire d'origine. Figure 22.

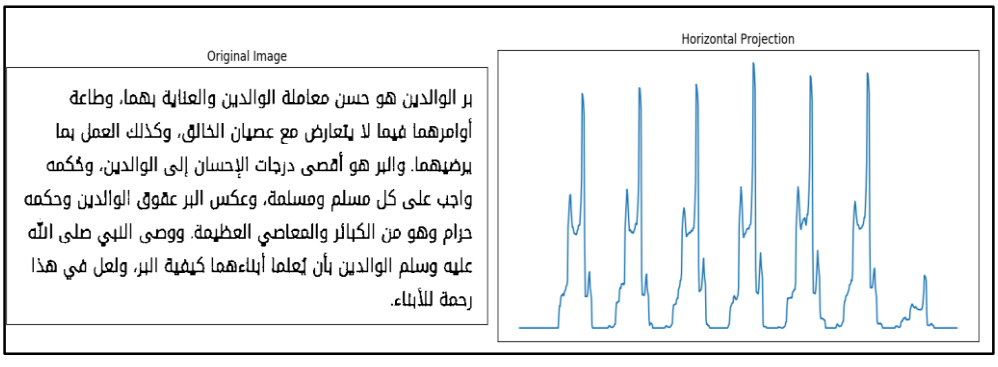


Figure 20 : (a) l'image de texte. (b) projection horizontale

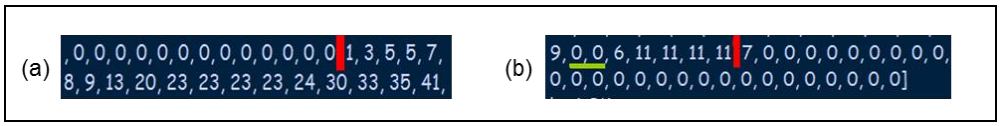


Figure 21 : (a) début de ligne. (b) fin de ligne

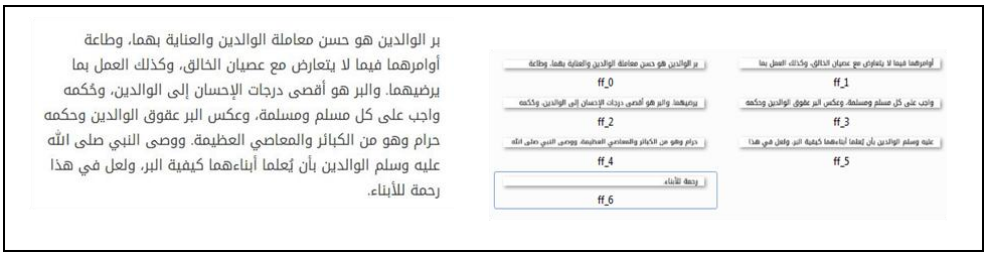


Figure 22 : résultats de la segmentation du texte en lignes par projection horizontale

2.1.2. Les problèmes rencontrés

Le plus grand défi pour cette méthode (projection) est le seuil choisi statiquement ; en effet, pour la segmentation du texte en plusieurs lignes, le seuil de trois pixels a été choisi statiquement après plusieurs tests. Mais il reste toujours un point faible pour les systèmes fonctionnant avec la projection horizontale, car pour quelques fontes, et quand la taille de police est très grande, l'espace entre le caractère et ses points diacritiques peut dépasser le seuil (4 pixels), et par la suite les points diacritiques sont considérés des lignes. Exemple de la figure 23.

Figure 23 : sur-segmentation des lignes

Le même problème peut être rencontré pour la segmentation des lignes en pseudo-mots. L'espace entre deux pseudo-mots peut être très large et dépasse le seuil de segmentation en mots, et par conséquences, les deux pseudo-mots sont considérés comme des mots.
La meilleure solution pour ces problèmes est de rendre le système capable de choisir automatiquement et d'une manière dynamique le seuil de segmentation.

2.2. Segmentation des lignes en pseudo-mots

2.2.1. Principe

Pour chaque ligne précédemment extraite, les mots et pseudo-mots (qui sont séparés par des espaces blancs) sont extraits dans cette étape en utilisant la technique de projection verticale. L'image binaire donnée (segment de ligne) est scannée, et pour chaque colonne, le nombre des pixels noirs est stocké dans un tableau de la même largeur que l'image. Le début

d’un pseudo-mot est indiqué par la fin d’une suite des zéros successifs. Du même principe, la fin d’un pseudo-mot est indiquée par le début d’une suite des zéros successifs. La figure 24

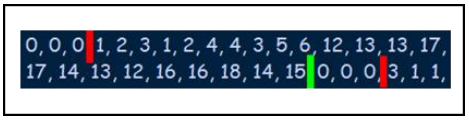


Figure 24 : (a) début (rouge) et fin (vert) du pseudo-mot.

Pour chaque pseudo-mot, et après avoir localisé et sauvegardé les débuts et les fins des zones de séparation, nous extrayons l’image sous forme des parties de la matrice de l’image binaire d’origine. La figure 25 montre un exemple d’une ligne de texte et ses pseudo-mots séparés.

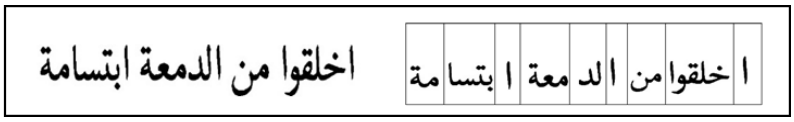


Figure 25: Segmentation d’une ligne en pseudo-mots par projection verticale

2.2.2. Les problèmes rencontrés

Le plus grand problème que l’on peut rencontrer dans cette étape est celui de chevauchement, montré dans la figure 26, où des parties de deux caractères différents partagent la même position horizontale. La projection verticale dans ce cas ne détecte aucune colonne vide entre les deux pseudo-mots et par conséquences les considère comme étant un seul pseudo-mot. Le bruit également peut simuler l’effet du chevauchement, c’est pour cela que nous insistons sur une bonne phase de prétraitements. Ainsi l’écriture italique peut donner même problème.

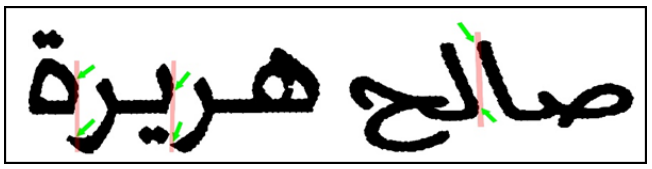


Figure 26 Problème de chevauchement

2.3. Segmentation des pseudo-mots en caractères

Après avoir segmenté le texte en lignes par la méthode de projection horizontale et les lignes en pseudo-mots par la méthode de la projection verticale, vient l’étape la plus importante et la plus décisive ; c’est la segmentation des pseudo-mots en plusieurs caractères.
Il s’agit dans cette étape de localiser les points de segmentation possibles entre les caractères connectés. Pour cela nous avons étudié et implémenté trois méthodes : Template-Matching, segmentation à base de seuil, projection verticale modifiée.

2.3.1. Template-Matching

Dans cette section, nous avons étudié et implémenté la méthode du Template-matching décrite dans [BS97'].

En observant un mot arabe, exemple de Figure 27, il se voit que les caractères sont connectés entre eux par une ligne de base. La méthode consiste à chercher le point de connexion entre le caractère et cette ligne base. Pour cela, le premier objectif doit être de déterminer la ligne base. Ensuite parcourir point par point la ligne juste en dessus de la ligne de base pour vérifier la coïncidence du voisinage du point courant avec la Template. Si le point satisfait la Template choisie, il est considéré un point de segmentation.

Figure 27 : connexion des caractères et la ligne de base

2.3.1.1. Estimation de la ligne de base

La ligne de base est la ligne horizontale contenant le nombre maximal de points noirs dans l'image binaire. Notre objectif est de trouver la première rangé de cette ligne (dans la direction verticale), cela est fait en calculant la valeur maximale de la projection horizontale de l'image (valeur de pic), puis on fixe un seuil de 70% de cette valeur. La première rangée de la ligne de base est la première rangée rencontrée (du haut vers le bas de l'image entière) dont le nombre de points noirs satisfait le seuil. Ainsi qu'on détermine la fin de la ligne de base qui est –à partir du début détecté précédemment- la première rangée présentant le nombre des points noirs inférieur au seuil. Comme résultat, on obtient le début et la fin de la ligne de base.

2.3.1.2. Détection des points de segmentation

La rangée juste au-dessus de la ligne de base est balayée de gauche à droite, et pour chaque point, le voisinage est testé avec la Template de la Figure 28 où p représente le point courant, 0 représente un point blanc et 1 un point noir, x n'est pas important, si les deux Y sont à 1, cela indique la présence de (-). Si le voisinage de P satisfait cette condition, alors le point P est un point de segmentation. Et le caractère peut être segmenté dans ce point verticalement sur la hauteur de la ligne de base. Dans le cas parfait c.à.d. l'image est d'une meilleure qualité, et le bruit est réduit au minimum, nous obtenons un résultat similaire à celui de la Figure 29.

X	X	X	Y	0	0	0
1	X	X	Y	0	0	0
X	1	X	0	0	0	0
X	1	1	P	0	0	0
1	1	1	1	1	1	1
X	X	X	X	X	X	X
X	X	X	X	X	X	X

Figure 28 : Template utilisée

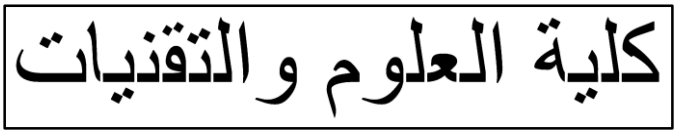


Figure 29 : Résultat de la segmentation par Template-Matching

2.3.1.3. Problème 1 : cas particuliers

Pour certains caractères arabes, l'angle qui joint le caractère à la ligne de base se trouve au-dessous du caractère lui-même. Ainsi que les caractères ayant les trous sur la ligne de base comme (م) (ص) (ط) (ص) (ض) (ظ) (و) (ه) (و) présentent un autre problème pour cette méthode de segmentation ; pour quelques fontes, le système considère l'intérieur du trou comme un point de segmentation. La Figure 30 illustre ces situations. Pour les caractères (ـب) (ـت) (ش) (س) (ـصـ) (ـضـ), la segmentation donne plus que 2 segments. La Figure 31 montre un exemple de la sur-segmentation.

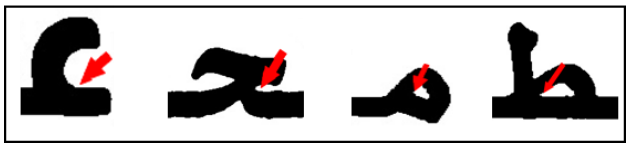


Figure 30 : exemples des cas particuliers

Figure 31 : Problème de sur-segmentation.

Ces points, bien qu'ils satisfassent la condition de la Template, ils ne doivent pas être pris comme des points de segmentation, car la coupure dans ces points va découper le caractère en deux parties ou plus.

2.3.1.4. Problème 2 : ligne de base

Parfois, la ligne de base n'est pas bien estimée, et cela revient à la non-continuité et au bruit sur la surface de la ligne de base, Figure 32.a ; par conséquences certains caractères ne sont pas segmentés ou ils sont sur-segmentés : Figure 32.b.

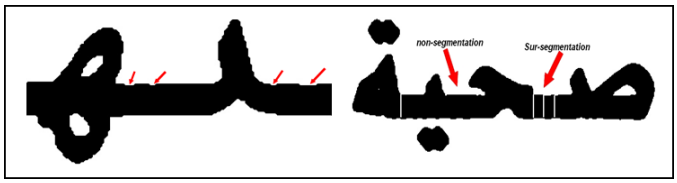


Figure 32 : (a) non continuité de la ligne de base. (b) fausse segmentation et sur-segmentation

2.3.2. Segmentation à base de seuil

Pour segmenter les mots arabes en caractères, la propriété fondamentale de la connectivité est décomposée. Comme indiqué précédemment, cette propriété dépend en grande partie de la ligne de base. Si une projection verticale est effectuée sur le mot (à savoir la somme

de chaque pixel noir à la verticale), le point de connexion affiche la moindre somme de la valeur moyenne (AV).

$$AV = \frac{\sum_{i=1}^{Nc} Xi}{Nc}$$

Par conséquent, chaque partie doit être segmentée montrant une valeur de somme inférieure à AV dans un autre caractère.

La première étape après l'acquisition de l'image binaire contenant le mot, est d'obtenir son histogramme vertical, puis il s'agit de calculer sa valeur moyenne : Figure 33.a. Puis retourner les indices des colonnes qui présentent des valeurs de projection inferieurs à la valeur moyenne, et l'image est mise à zéro (blanc) dans ces indices. Figure 33.b.

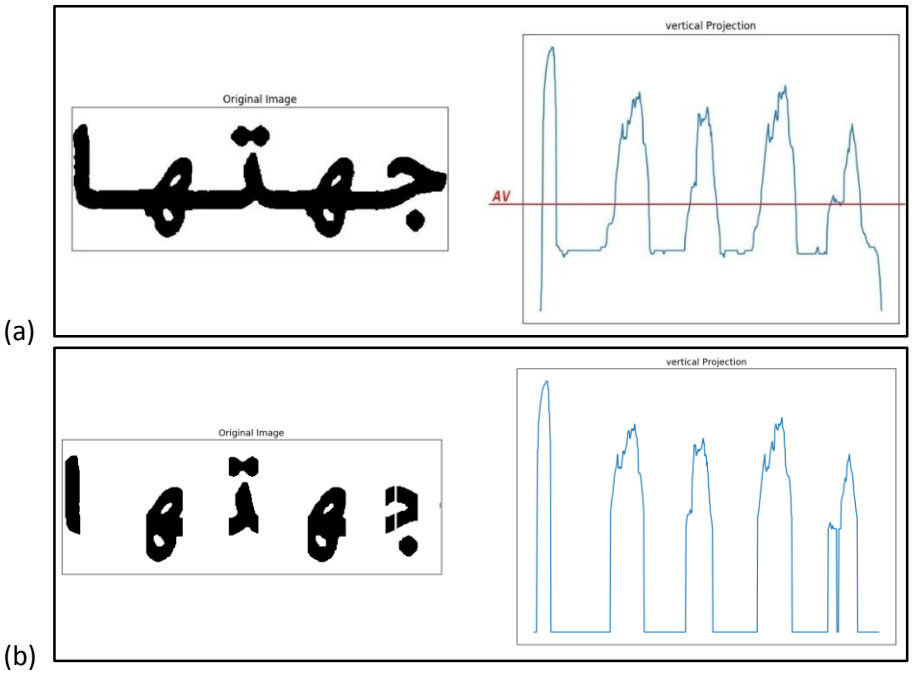


Figure 33:(a) la valeur moyenne de la projection verticale d'un mot arabe (b) séparation des caractères

Pour obtenir les points de segmentation, il suffit d'appliquer la méthode de segmentation des lignes en pseudo-mots (projection verticale).

2.3.3. Projection verticale modifiée

Cette troisième méthode consiste à éliminer la ligne de base pour créer le vide entre les caractères et ensuite appliquer une projection verticale simple pour localiser les points de segmentation.

2.3.3.1. Estimation de la ligne de base

Pour détecter la ligne de base, on utilise le même principe expliqué précédemment dans la méthode de Template-Matching.

2.3.3.2. Elimination de la ligne de base et détection des points de segmentation

Tous les points de la ligne de base sont mis à zéro (blanc). Maintenant que nous avons le mot avec la ligne de base éliminée, nous pouvons appliquer la méthode de la projection verticale pour détecter les points de segmentation. Nous sauvegardons les positions des débuts et fins des caractères en détectant les vides entre les caractères dans le mot sans ligne de base, puis nous sauvegardons les images des caractères segmentés en utilisant le mot original et les positions sauvegardés. Figure 34.

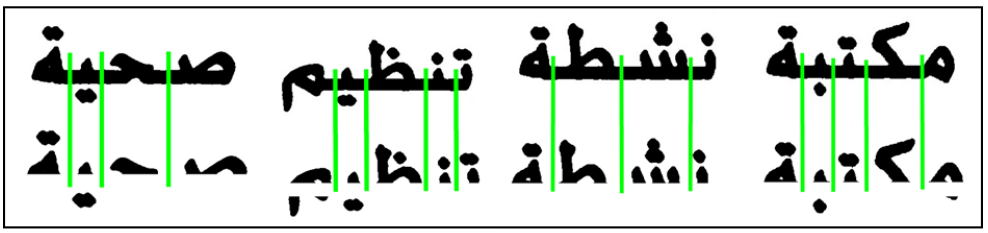


Figure 34 : Segmentation par élimination de la ligne de base

2.3.3.3. Problème 1 : élimination de la ligne de base

Parfois (mais très rarement) il y a des mots où la ligne de base n'est pas complètement éliminée, exemples de la Figure 35, parce que la fin de ou le début de la vraie ligne de base ne satisfait pas le seuil de 70% de nombre maximal de points noir par une ligne. Cet effet empêche d'appliquer la projection verticale parce que le vide n'a pas été créé correctement.

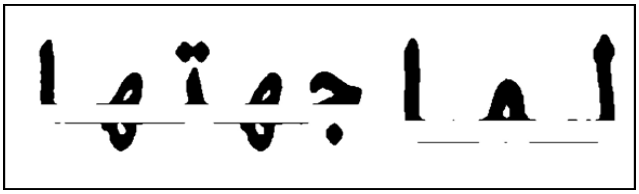


Figure 35 : ligne de base non complètement éliminée

2.3.3.4. Problème 2 : Sur-segmentation.

Le problème de la sur-segmentation est toujours présent. Pour les caractères (بـ) (تـ) (شـ) (سـ) (صـ) (ضـ), chacun de ces caractères est segmenté en plus que deux parties. La résolution de ce problème est laissée à la phase de reconnaissance.

2.3.4. Méthode basée sur le contour

Dans cette section, nous étudions la méthode décrite dans [CHKY96].

Cette méthode est destinée à la segmentation du texte manuscrit, mais nous l'avons adapté pour segmenter le texte imprimé. L'idée de base de cette méthode est de détecter le contour du mot, puis extraire le contour supérieur avec les codes de Freeman (F3 F5 F1), puis localiser les points de segmentation qui sont considérés les minimums locaux du contour supérieur. Le problème majeur de cette méthode est la difficulté de détection de la ligne base,

vue que l'écriture manuscrite diffère d'une personne à autre. Nous avons implémenté cette méthode, le teste avec un texte manuscrit à donner un taux d'erreur de 68% alors que pour le texte imprimé en multi-fonte, le taux de réussite était de 86.19%. Plus de détails sur cette méthode sont découverts dans le chapitre suivant.

2.4. Résultats

L'évaluation des méthodes de segmentation, est faite sur une base de données composée de **1330 mots** arabe imprimés contenant au total **4892 caractères** écrits avec 27 fontes différentes.

La segmentation du texte en plusieurs lignes par la méthode de projection horizontale a donné un taux d'erreur de **0%.** La projection verticale pour la segmentation des lignes en pseudo-mots a également donné un résultat parfait avec au taux d'erreur de **0%**.

La méthode du Template-matching a réussi à segmenter correctement **3907** caractères, avec un taux d'erreur de **20.14%.** La deuxième méthode - Projection verticale modifiée – a réussi à segmenter **3876** caractères avec un taux d'erreur de **20.76%.** Pour la troisième méthode, celle à base de seuil, **3779** caractères ont été correctement segmentés et le taux d'erreur était **22.75%,** alors que pour la méthode basée sur le contour, le taux d'erreur était **13.81%.**

Le tableau 2 résume les résultats obtenus dans cette évaluation.

	Nombre de caractères bien segmentés	Taux de réussite %	Taux d'erreur %
Template-Matching	3907	79.86	20.14
Projection à base de seuil	3799	77.25	22.75
Projection Verticale	3876	79.23	20.76
Contour	4216	86.19	13.81

Tableau 2 : résultat de segmentation

Conclusion

Nous avons vu dans ce chapitre une étude de quatre méthodes de segmentation de texte arabe. D'après les résultats du tableau 2, il se voit que la méthode du contour et celle du Template-Matching, donnent les meilleurs résultats.

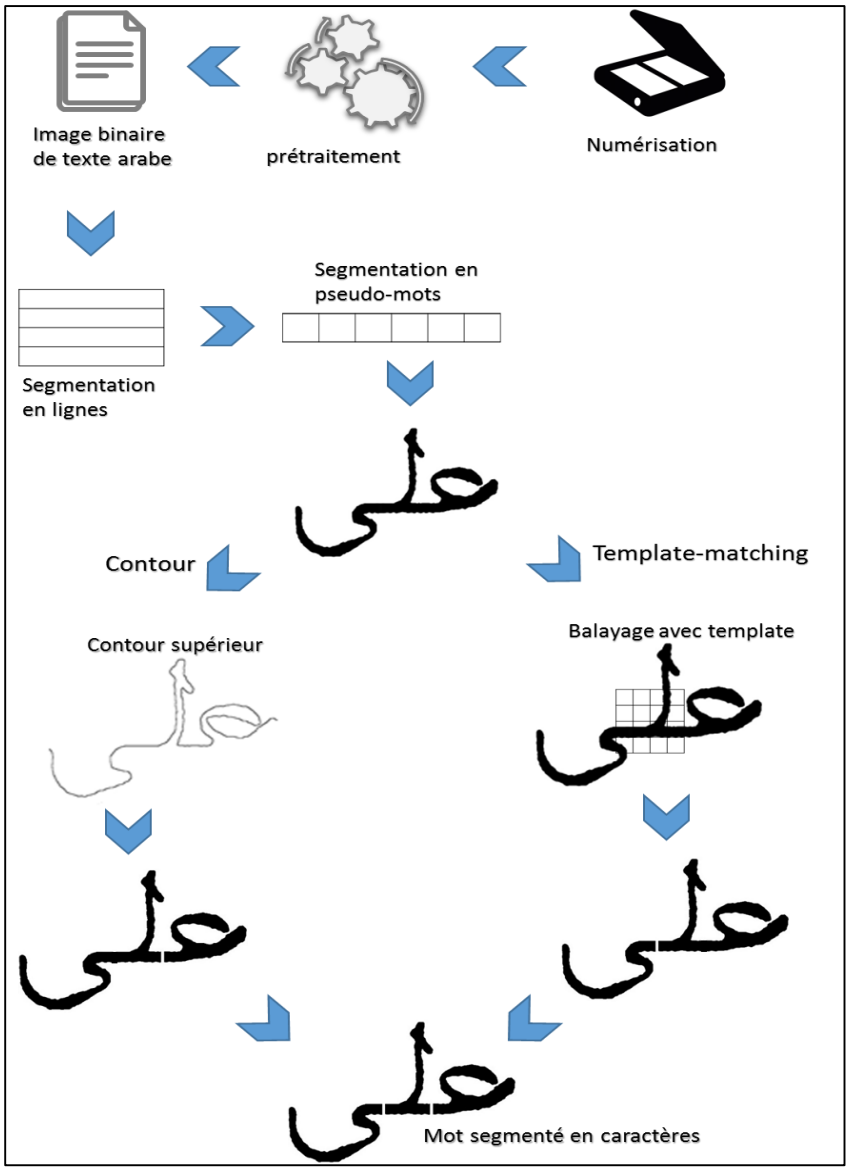
CHAPITRE 3. Contribution à la segmentation du texte arabe imprimé

Introduction

Dans ce chapitre, nous présentons notre contribution à la segmentation du texte arabe en caractères, en proposant des améliorations pour la méthode de Template-Matching. Nous étudions également d'une manière détaillée la méthode basée sur le contour, nous révélons ses lacunes et nous proposons des solutions. Le choix de ces deux méthodes vient du fait qu'elles ont donné les meilleurs résultats, pour notre base de données, (voir tableau 2), et qu'il y a une grande possibilité d'augmenter le taux de réussite de la segmentation pour ces deux méthodes.

Notre méthode proposée est sous forme d'une méthode hybride et robuste, qui se serve de l'analyse du contour et du balayage par fenêtre glissante (Template-Matching).

L'image numérique acquise subit des opérations de prétraitement (binarisation, réduction de bruit…), puis elle est passée pour le système de segmentation. Le texte dans l'image est segmenté premièrement en plusieurs lignes, puis chaque ligne se segmente en plusieurs pseudo-mots. Chaque pseudo-mot est segmenté en utilisant la méthode de template matching pour extraire les caractères descendants. Le reste du pseudo-mot est segmenté en utilisant la méthode du contour. Le schéma suivant résume les étapes du processus de segmentation proposé :



3.1. Acquisition

Le document qui contient le texte arabe imprimé est d'abord numérisé avec une résolution de 700 ppp, il est chargé puis converti en niveaux de gris.

3.2. Prétraitement

3.2.1. Amélioration de la résolution

Dans certains cas, l'image acquise est de faible résolution, ce qui empêche de distinguer la continuité des pixels. Pour cela, nous proposons d'augmenter la résolution de l'image à 2 fois la résolution initiale. Il faut noter que cette phase doit précéder l'étape de binarisation.

3.2.2. Binarisation

Pour convertir l'image en noir et blanc, nous proposons d'utiliser une simple binarisation (seuillage simple) qui consiste à comparer les niveaux de gris de chaque point de l'image avec un seuil -que nous avons fixé à 180- pour décider à quelle des deux classes (noir/blanc) appartient ce point. [03]

3.2.3. Opérations morphologiques

3.2.3.1. Définitions

La transformation morphologique modifie la valeur d'un pixel de l'image en fonction de la valeur de ses voisins. Pour cela, on utilise un élément structurant, qui est un masque binaire. Il permet de prendre en compte le voisinage du pixel.
Exemple d'un élément structurant à 4-connexité :

0	1	0
1	1	1
0	1	0

Les traitements morphologiques sont définis à partir de deux opérations de base qui sont l'érosion et la dilatation. [04]

Erosion

Cette opération amincit les bordures de l'objet de premier plan. Le noyau se glisse sur l'image. Un pixel dans l'image originale (soit 1 ou 0) ne sera considéré « 1 » que si tous les pixels sous le noyau, dont il est le centre, sont à « 1 », sinon il est érodé (mis à zéro). [04]

Dilatation

C'est l'inverse de l'érosion. Ici, un élément de pixel est «1» si au moins un pixel sous le noyau est «1». Donc, il augmente la taille de l'objet de premier plan. Normalement, la dilatation ne doit être appliquée qu'après une érosion, pour ne pas agrandir le bruit. [04]

Ouverture

C'est juste un autre nom d'une érosion suivie d'une dilatation. Elle est très utile pour réduire le bruit dans les images, parce que, l'érosion élimine les petits points de bruits, mais elle amincit également l'objet principal. Nous le dilatons à sa forme initiale. Comme les points de bruit sont partis, ils ne reviendront pas. [04]

3.2.3.2. Application

Souvent, le bruit apparait dans l'image acquise, et cela est dû à la qualité de la numérisation. Ce bruit est sous forme des points additionnels ne faisant pas partie du texte ; pour éliminer ces points et ne garder que le texte sur l'image, nous proposons d'utiliser l'opérateur morphologique « ouverture » qui consiste à appliquer une érosion morphologique suivie d'une dilatation avec un même élément structurant de 8-connexité. Cela permet dans la première étape d'éroder le texte et par conséquences éliminer les points de bruit, et dans la

deuxième étape de rendre le texte à son épaisseur d'origine. La figure 36 montre le résultat des deux dernières opérations.

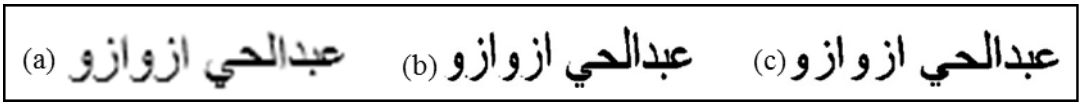


Figure 36 : (a) l'image originale. (b) : l'image binaire. (c) : l'image après l'ouverture morphologique

3.3. Segmentation des caractères arabes imprimés

3.3.1. Méthode basée sur Template-Matching

La méthode de Template-Matching implémentée dans le chapitre précèdent a montré des lacunes au niveau de la segmentation, et un faible taux de segmentation. Dans cette section, nous proposons des améliorations pour cette méthode. Les résultats de nos améliorations sont montrés et discutés dans le chapitre 4.C.

La première amélioration est une nouvelle Template. Après avoir essayé des différentes templates, certaines donnent des meilleurs résultats (taux d'erreur moins de 10%), mais seulement pour des fontes particulières (MCS-Masahif-S_U-normal, ACS-Almas, A-Mosalas…). Pour une bonne segmentation de maximum de fontes, nous proposons d'utiliser la Template de la figure 37 qui donne des meilleurs résultats par rapport à la Template précédente.

x	x	x	x	x	x
1	x	x	0	0	0
1	1	x	0	0	0
x	1	1	P	0	0
1	1	1	1	1	1

Figure 37 : Template proposée

Ainsi que pour les problèmes décris dans le chapitre précèdent (Chapitre 2.D.1), nous proposons les solutions suivantes :

3.3.1.1. Solution 1 :

Pour éviter de segmenter le caractère en deux parties (cas de la figure 30), nous proposons d'ignorer le point P1 qui satisfait la condition de la Template, et le point de segmentation est choisi étant –à partir du point P1- le premier point à droite qui n'a pas de point noir au-dessus et qui appartient à la ligne de base. Figure 38. Pour le problème de la sur-segmentation, il sera résolu dans l'étape de la reconnaissance.

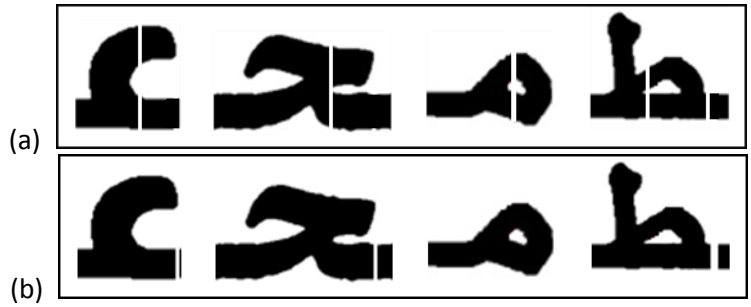


Figure 38 : Résultat de la segmentation : (a) avant l'amélioration. (b) après l'amélioration

3.3.1.2. Solution 2 :

Pour garantir la segmentation de tous les caractères, nous proposons de chercher les points de segmentation non seulement sur la première rangée au-dessus de la ligne de base mais également sur les deux niveaux supérieurs et inférieurs, comme montre la figure 39. On est sûr que le point de segmentation est détecté dans un de ces 5 niveaux. Notons que l'image est d'une résolution élevée, 5 pixels sont ignorés par rapport à la hauteur de la ligne de base. Pour le cas de la sur-segmentation, il ne pose pas un grand problème quand les parties sur-segmentées ne contiennent aucune information.

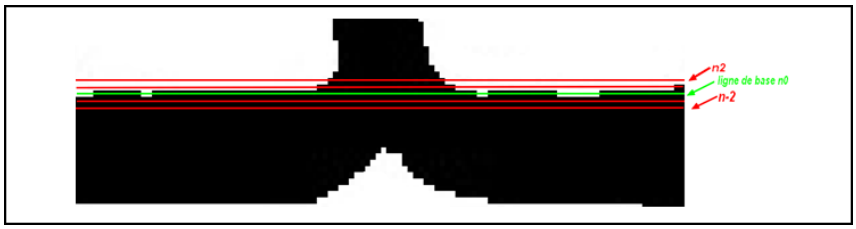


Figure 39 : Les cinq niveaux de balayage de la ligne de base

3.3.2. Méthode basée sur le contour

3.3.2.1. Généralités

Les bords sont la caractéristique de l'objet qui décrit parfaitement sa forme. En fait la vision humaine dépend fortement des bords des objets, car ils nous permettent de distinguer entre les objets sans avoir recours aux autres caractéristiques telles que les couleurs, luminance… Pour la vision de la machine (imagerie) ces bordures sont dites les contours.

La détection de contour est un champ de recherche qui appartient au traitement d'image et à la vision par ordinateur, particulièrement dans le domaine de l'extraction de caractéristiques.

La détection du contour s'agit de localiser les points d'une image numérique qui correspondent à un changement brutal de l'intensité lumineuse. Ces changements de propriétés de l'image traduisent en général des événements importants ou des changements dans les propriétés des objets. Ils incluent des discontinuités dans la profondeur, dans l'orientation d'une surface, et dans l'éclairage d'une scène.

La détection des contours d'une image réduit de manière significative la quantité de données et élimine les informations qu'on peut juger moins importantes, tout en préservant les propriétés

structurelles importantes de l'image. Il existe un grand nombre de méthodes de détection de contour mais la plupart d'entre elles peuvent être regroupées en deux catégories. La première recherche les extremums de la dérivée première, en général les maximums locaux de l'intensité du gradient. La seconde recherche les annulations de la dérivée seconde, en général les annulations du Laplacien ou d'une expression différentielle non linéaire.

3.3.2.2. Principe de la segmentation

Pour segmenter les pseudo-mots en caractères individuels, nous proposons d'extraire le contour du pseudo-mot en utilisant le détecteur de Canny [02], ensuite extraire le contour supérieur en utilisant la chaine de Freeman [H68], et le parcourir pour localiser les points de segmentation dans ses minimums locaux qui appartiennent à une zone précise, et finalement séparer les caractères.

Le contour inferieur ne contient aucune information concernant la connexion des caractères avec la ligne de base. Tous les caractères de presque toutes les fontes sont connectés à la ligne de base en haut. Donc c'est la partie supérieure du contour qui peut contenir les points de segmentation.

3.3.2.3. Outils

Détecteur de Canny

Le détecteur de Canny [02] est une méthode analytique de détection de contour, elle est basée sur la recherche des maximums locaux du gradient.

Le détecteur de Canny (ou filtre de Canny) est le plus utilisé aujourd'hui pour la détection des contours. L'algorithme a été conçu par John Canny en 1986 pour satisfaire trois critères clairement explicités :

- Bonne détection : faible taux d'erreur dans la signalisation des contours
- Bonne localisation : minimisation des distances entre les contours détectés et les contours réels.
- Clarté de la réponse : une seule réponse par contour et pas de faux positifs

La détection de contour par Canny passe par les étapes suivantes :

1) Filtrer tout bruit. Le filtre gaussien est utilisé pour cet objectif. Un exemple d'un noyau gaussien de taille = 5 qui pourrait être utilisé est présenté ci-dessous :

$$K = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

2) Trouvez le gradient d'intensité de l'image. Pour cela, on suit une procédure analogue à Sobel [05] :

Appliquer une paire de masques de convolution (dans les directions x et y) pour obtenir la première dérivée dans la direction horizontale (G_x) et la direction verticale (G_y) :

$$G_x = \begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix} \qquad G_y = \begin{matrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{matrix}$$

Trouvez la valeur et la direction du gradient avec :

$$G = \sqrt{G_x^2 + G_y^2}$$

$$\theta = \arctan(\frac{G_y}{G_x})$$

La direction du gradient est toujours perpendiculaire aux bords, elle est arrondie à l'un des quatre angles représentant des directions verticales, horizontales et deux diagonales. (0°, 45°, 90° ou 135°)

3) Suppression des non-maximums

Après avoir obtenu la valeur et la direction du gradient, une analyse complète de l'image est effectuée pour supprimer les pixels indésirables qui ne constituent peut-être pas le bord. Pour cela, chaque pixel est vérifié s'il s'agit d'un maximum local dans son voisinage et dans la direction du gradient, voir l'exemple de la figure 40 :

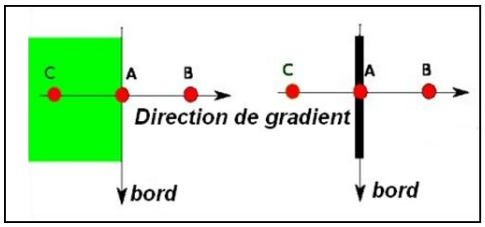


Figure 40 : Canny - suppression des non-maximums

Le point A est sur le bord (en direction verticale). La direction du gradient est normale au bord. Les points B et C sont dans la direction du gradient. Le point A est vérifié avec les points B et C pour voir s'il forme un maximum local. Si c'est le cas, il est considéré pour la prochaine étape, sinon, il est supprimé (mis à zéro).

Le résultat que vous obtenez est une image binaire avec des bords minces.

4) Seuil d'hystérésis :

Cette étape décide les vrais et les faux bords. Pour cela, nous avons besoin de deux valeurs de seuil, minVal et maxVal. Tous les bords avec un gradient d'intensité supérieur à maxVal sont sûrs d'être des bords et ceux en dessous de minVal sont sûrs d'être non-bords, donc jetés. Ceux qui se situent entre ces deux seuils sont classés dans les bords ou les non-bords en fonction de leur connectivité. S'ils sont connectés à des pixels de bord, ils sont considérés comme faisant partie des bords également. Sinon, ils sont éliminés.

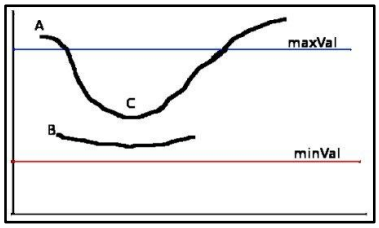


Figure 41 : Canny - Seuil d'hystérésis

Dans l'exemple de figure 41 le bord A est au-dessus du maxVal, donc considéré comme un vrai bord. Bien que le bord C soit inférieur à maxVal, il est connecté au bord A, de sorte que également considéré comme un vrai bord et on obtient cette courbe complète (AC). Mais le bord B, bien qu'il soit au-dessus de minVal et soit dans la même région que celui du bord C, il n'est pas connecté à un "vrai bord", donc il est mis au rebut. Il est donc très important de bien choisir minVal et maxVal pour obtenir le bon résultat.
Cette étape supprime également les petits bruits en supposant que les bords sont des lignes longues. Donc, ce que nous obtenons finalement, c'est des bords forts dans l'image.
La figure 42 montre un exemple de détection de contour par Canny.

Figure 42 : exemples de détection de contour par Canny

Code de Freeman

Le code (ou la chaine) de Freeman [H68] est une méthode de recherche de contour basé sur une fenêtre de 3*3. La réussite de la méthode dépend de deux facteurs : le premier est le sens du passage, c.à.d. le prochain pas est-il dans le sens des aiguilles de la montre ou dans le sens inverse ? Le deuxième facteur est le point de départ. La chaine de Freeman est une représentation qui se compose des séries de numéros de 0 à 7, chaque numéro représente la direction du pixel suivant connecté dans la fenêtre 3*3 comme indiqué dans la Figure 43. La coordonnée du pixel suivant est calculée en fonction de l'addition et de la soustraction de colonnes et de lignes par 1.

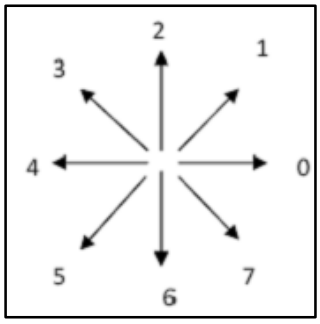


Figure 43 : les 8 directions de Freeman

Selon ces codes, la position de pixel suivante peut être obtenue en se référant au tableau 3. La figure 43 contient les sept codes et donne également les positions de pixels suivants en coordonnées x et y :

Code	Ligne suivante	Colonne suivante
0	**X**	**Y+1**
1	**X-1**	**Y+1**
2	**X-1**	**Y**
3	**X-1**	**Y-1**
4	**X**	**Y-1**
5	**X+1**	**Y-1**
6	**X+1**	**Y**
7	**X+1**	**Y+1**

Tableau 3 : detection du pas suivant en fonction des codes de Freeman

3.3.2.4. Segmentation

Division du mot verticalement

L'objectif de cette étape est de diviser le mot verticalement en 3 zones : une zone supérieure, une zone médiane, et une autre inferieure. Cette opération a pour but de minimiser l'espace de la recherche des points de segmentation dans la zone médiane pour éviter les problèmes de la ligne de base, et pour que la segmentation soit indépendante de la ligne de base.

La zone médiane se compose de la ligne de base plus un tiers de la ligne de base en haut et en bas. La Figure 44 montre un exemple de la détection de cette zone.

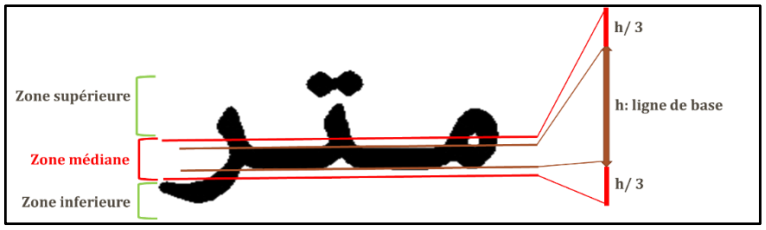


Figure 44 : La répartition verticale d'un mot arabe

Détection du contour

Pour détecter le contour du mot (image binaire), nous avons utilisé le détecteur de Canny décrit précédemment. Le choix d'utiliser Canny est à cause de sa robustesse et sa bonne performance en extraction du contour. Figure 45.

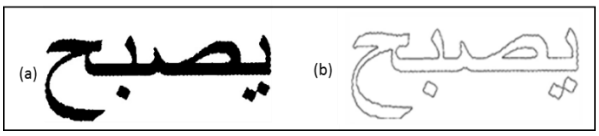


Figure 45 : (a) mot arabe sur image binaire. (b) le contour correspondant

Résolution du problème de chevauchement

Pour les méthodes implémentées précédemment, le chevauchement des caractères est l'un des problèmes qui rendent la segmentation des mots arabes en caractères une tâche complexe, voire impossible pour quelques fontes. Le chevauchement est détecté quand deux caractères partagent une même position verticale, ils se voient comme un seul caractère, exemple de la Figure 46. Vu que la projection verticale ne peut pas séparer les pseudo-mots en chevauchement dans l'étape de la segmentation des lignes, ce problème reste à la dernière étape de segmentation.

Le traçage du contour du mot entraîne un contour fermé pour chacun des pseudo-mots. Le problème de segmentation se réduit donc à la segmentation de chaque pseudo-mot si celui-ci contient plus d'un caractère.

Figure 46 : chevauchement vertical des caractères

La figure 46 montre une phrase ayant 5 chevauchements, pour séparer cette phrase en pseudo-mots, il suffit de tracer le contour de la phrase, ce qui donne 9 contours fermés différents Figure 47.a. Pour chaque contour en intersection avec la zone médiane (ignore les contours des points diacritiques), on construit une image binaire contenant le pseudo-mot correspondant avec ses points diacritiques Figure 47.b.

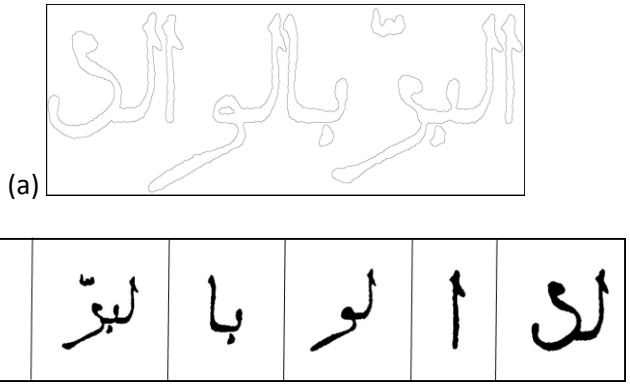


Figure 47 : (a) contour, (b) séparation des pseudo-mots en chevauchement

Le résultat de cette étape est la séparation de tous les pseudo-mots en chevauchement.

Extraction du contour supérieur

Pour chaque pseudo-mot résultant de l'étape précédente, on trace le contour avec le détecteur de Canny. Ce contour se construit de deux parties, une partie supérieure, et une partie inférieure, et vu que les caractères arabes sont connectés à la ligne de base sur le contour supérieur (Figure 48). On extrait seulement celui-ci.

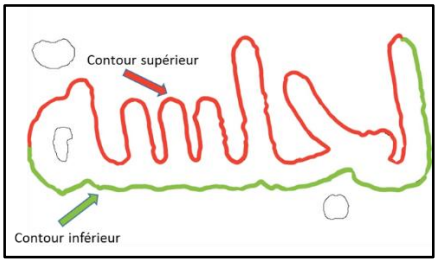


Figure 48 : Le contour supérieur et le contour inferieur d'un mot arabe

En premier temps, on choisit deux points sur le contour ; le point le plus à droite, et celui le plus à gauche. En commençant par un des deux on construit une **chaine de Freeman** pour arriver au deuxième point. Il faut noter que le premier pas doit être vers le haut, c.à.d. une des directions (1, 2, 3) pour éviter d'extraire le contour inférieur. La Figure 49 montre un exemple.

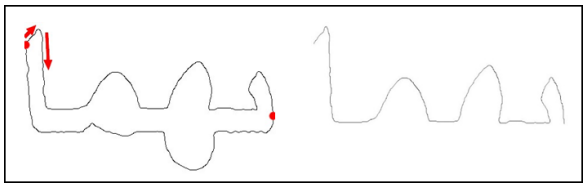


Figure 49 : Extraction du contour supérieur

Détection des points de segmentation

Rappel mathématique [06] :

Une fonction f : D → R possède un minimum local au point m∈D s'il existe un intervalle $I =]m - \delta, m + \delta[$ contenu dans D sur lequel f(m) ≤ f(x). La valeur f(m) s'appelle un minimum local de f et m l'argument de ce minimum local.

Pour localiser les points de segmentation potentiels, le contour supérieur est analysé, et ses minimums locaux sont extraits, Figure 50.

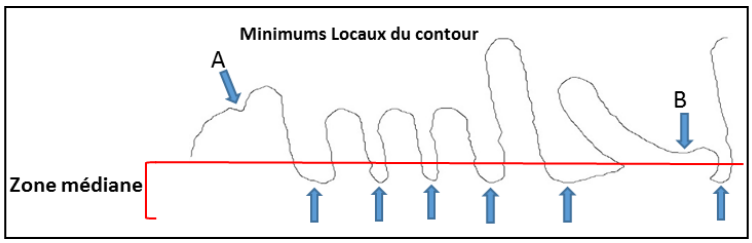


Figure 50 : Les minimums locaux du contour supérieur

On définit **une zone de segmentation** comme étant une partie du contour, continue horizontalement, et contenant plusieurs minimums locaux successifs.

Les zones A et B sur la Figure 50, forment des minimums locaux pour contour, pourtant ils ne doivent pas être considérés des zones de segmentation. La solution pour cette situation, est de ne considérer que les minimums qui appartiennent à la zone médiane du mot.

Après avoir détecté les zones de segmentation, on choisit le milieu de chaque zone comme étant le point de segmentation. Résultat de la segmentation dans la figure 51.

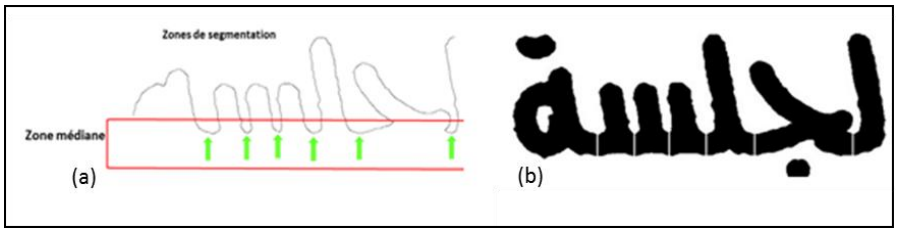


Figure 51 : (a) zones de segmentation. (b) résultat de la segmentation

3.3.2.5. Problème 1

Pour quelques fontes, les points ou signes de voyellation d'un caractère peuvent apparaitre avant ou après le caractère (de droite à gauche), c'est-à-dire, un diacritique ne se trouve pas complètement sur le caractère, exemple de Figure 52. Ceci va poser un problème lors de la détection des points de début et fin du contour supérieur. On ne pourra jamais trouver le chemin entre le point de départ et celui d'arrivée. Car un point fait partie du mot et l'autre fait partie du diacritique.

Figure 52 : problème de détection de contour supérieur

3.3.2.6. Solution 1

Vu que les points et les signes diacritiques ne font pas partie du processus de segmentation, l'idée est de les éliminer et travailler uniquement sur le corps du mot. Pour cela,

nous avons tracé les contours du mot et prendre uniquement le contour le plus grand et qui est en intersection avec la zone médiane du mot, ce qui correspond au corps du mot. Cette opération remplie également les trous du mot. Voir exemple de la Figure 53.

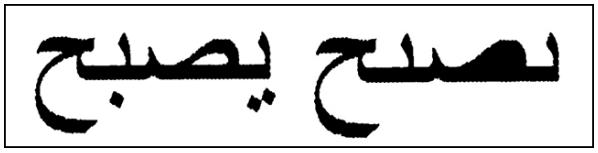


Figure 53 : élimination des points

3.3.2.7. Problème 2 : coupure du contour

La détection de contour n'est pas toujours parfaite, le bruit peut causer une coupure dans contour ; Figure 54. Par conséquences, l'extraction du contour supérieur ne peut pas être faite correctement.

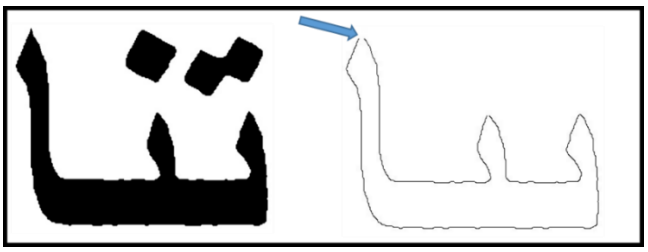


Figure 54 : coupure de contour

3.3.2.8. Solution 2

Pour que le contour puisse se fermer, nous proposons d'éliminer le bruit en appliquant une érosion morphologique suivie d'une dilatation avec des noyaux différents. Après ces deux traitements, nous pouvons extraire le contour supérieur du mot facilement en utilisant la chaine de Freeman.

3.3.2.9. Problème 3

Dans le cas parfait, il y a une seule zone de segmentation entre deux caractères, mais souvent, le bruit ou la mauvaise qualité de l'image peuvent donner naissance à plusieurs zones de segmentation, ce qui pose le problème de la sur-segmentation, où la ligne de base est segmentée en plusieurs parties de silence qui ne contiennent aucune information. Exemple de la figure 55.

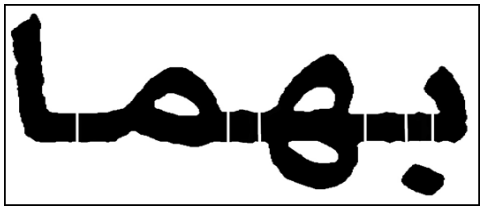


Figure 55 : sur-segmentation d'un mot arabe

3.3.2.10. Solution 3

Pour éliminer la répétition des points de segmentation, pour chaque point on détecte deux limites : à gauche et à droite du point. A partir du point de segmentation, la limite est la première colonne rencontrée qui a au moins un pixel noir dans la zone supérieure du mot. Et s'il n'y a aucune limite (cas du dernier caractère ou du premier caractère), le vide est considéré la limite. Figure 56.

Entre deux caractères (deux limites), nous proposons de ne valider qu'un seul point de segmentation, les autres sont ignorés. Vu que le texte arabe s'écrit de droite à gauche, le premier point à droite est validé, mais ce choix n'influe pas sur la segmentation.

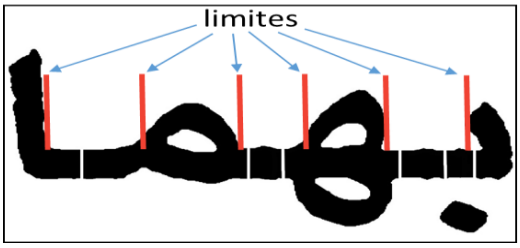


Figure 56 : élimination des répétitions

3.3.2.11. Problème 4

Pour quelques caractères (ـجـ ـحـ ـخـ ـعـ ـغـ ـعـ ـغـ ـكـ), le point de segmentation apparait au-dessous du corps du caractère, par conséquence, le caractère n'est pas complètement segmenté. Figure 57.

Figure 57 : cas particuliers

3.3.2.12. Solution 4

Pour remédier à ce problème, le point de segmentation est décalé à gauche ou à droite, entre ses deux limites, jusqu'à ce qu'il n'a aucune partie du caractère au-dessus. La figure 58 montre quelques résultats de la segmentation :

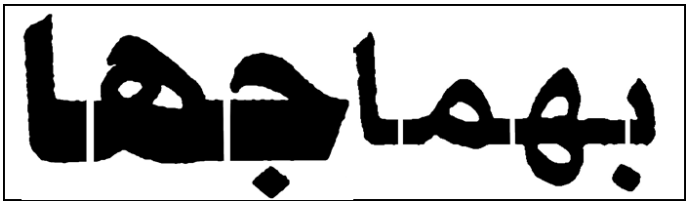


Figure 58 : résultat de la segmentation

3.3.2.13. Problème 5

Le 5ème défi de cette méthode est qu'elle n'est pas capable de segmenter les caractères descendants à la fin du pseudo-mot. On parle principalement du caractère (ى), et le caractère

(ر) dans certaines fontes. Vu que le contour (Figure 59) ne présente pas des zones de segmentation entre ce caractère descendant et le caractère qui le précède. La méthode basée sur le contour que nous avons décrite précédemment ne pourra pas segmenter ce type de caractère.

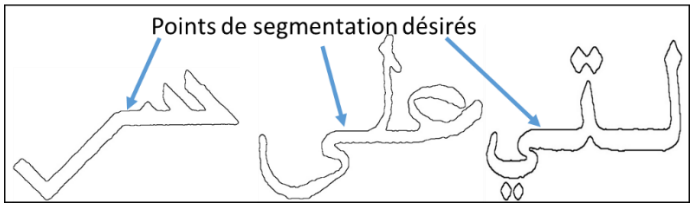


Figure 59 : caractères descendants

3.3.2.14. Solution 5

Comme solution pour cette situation, nous proposons d'utiliser la méthode du Template-Matching décrite précédemment dans le chapitre (CHAPITRE2.D) pour segmenter les caractères descendants à la fin du pseudo-mot.

Pour cela nous proposons la Template de la Figure 60.a. Cette méthode peut donner des résultats similaires à ceux de la Figure 60.b

(a)

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	P	1	1	1
0	0	0	1	1	1	1	1
0	0	0	1	1	1	1	1
0	0	0	1	1	1	1	1
0	0	0	1	1	1	1	1

(b)

Figure 60: (a) Template utilisée, (b) résultat de segmentation des caractères descendants

Premièrement le pseudo-mot est balayés avec la template, s'il y a des caractères descendants, ils sont segmentés et extraits. Le reste du corps de pseudo-mot est ensuite segmenté en utilisant la méthode de contour. Cet ordre de segmentation empêche la sur-segmentation.

Cette méthode est implémentée et testée avec les autres méthodes sur un corpus unifié et les résultats sont montrés et discutés dans le chapitre 4.C.

Conclusion

Dans ce chapitre nous avons présenté notre contribution à la segmentation du texte arabe imprimé en caractères. Nous avons proposé des solutions pour les lacunes de la méthode de Template-Matching. Ainsi que nous avons proposé un processus hybride de segmentation, basé sur le contour du texte et sur le balayage par fenêtre glissante (Template-matching), et nous avons traité les différents problèmes qu'un système de segmentation peut rencontrer.

CHAPITRE 4. Etude Expérimentale

Introduction

Dans ce chapitre, nous présentons la partie pratique de ce travail. L'environnement de développement Python-OpenCv, et le corpus unifié que nous avons construit pour avoir une comparaison claire entre les différents méthodes de segmentation que nous avons implémenté et qui sont décrites dans le chapitre 3 et le chapitre 4. Nous présentons ensuite en détails les résultats de différentes étapes de segmentation.

4.1. Environnement de développement

Python est un langage de programmation, qui est devenu très populaire en peu de temps principalement en raison de sa simplicité et de sa lisibilité de code. Il permet au programmeur d'exprimer ses idées dans moins de lignes de code sans réduire la lisibilité. Par rapport à d'autres langues comme C/C++, Python est plus lent. Mais une autre caractéristique importante de Python est qu'il peut être facilement étendu avec C/C ++. Cette fonctionnalité aide à écrire des codes intensifs en C/C++ et ce code peut être utilisé comme un module de Python. Cela donne deux avantages : d'abord, le code est aussi rapide que le code C/C++ original (puisqu'il s'agit du code C++ actuel en cours d'exécution) et, d'autre part, il est très facile de coder en Python. C'est comme cela que OpenCV-Python fonctionne, c'est un Python autour de l'implémentation originale de C++.

Et le soutien de NumPy rend la tâche plus facile. NumPy est une bibliothèque hautement optimisée pour les opérations numériques. Il donne une syntaxe MATLAB. Toutes les structures de tableau OpenCV sont converties en-et-à partir de tableaux NumPy. Donc, quelle que soit l'opération que vous pouvez faire à NumPy, vous pouvez la combiner avec OpenCV, ce qui augmente le nombre d'armes dans votre arsenal. En plus, plusieurs autres bibliothèques comme SciPy, Matplotlib qui supporte NumPy peuvent être utilisées avec ceci. En plus, plusieurs autres bibliothèques comme SciPy, Matplotlib qui supporte NumPy, peuvent être utilisées avec ceci. Ainsi, OpenCV-Python est un outil approprié pour le prototypage rapide de problèmes de vision par ordinateur.

4.1.1. Python

Python est un environnement de programmation objet, multi paradigmes et multi plateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet. Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire par ramasse-miettes et d'un système de gestion d'exceptions ; il est ainsi similaire à Perl, Ruby, Scheme, Smalltalk et TCL. Il vient avec plusieurs versions, mais les plus utilisées sont 3.5 et la **version 2.7** que nous avons utilisée dans ce travail.

Le langage Python est placé sous une licence libre et fonctionne sur la plupart des plates-formes informatiques, de Windows à Unix avec notamment GNU/Linux en passant par macOS, ou encore Android, iOS, et aussi avec Java ou encore .NET. Il est conçu pour optimiser la productivité des programmeurs en offrant des outils de haut niveau et une syntaxe simple à utiliser. [07]

4.1.2. Bibliothèques

Les bibliothèques suivant sont les bibliothèques que nous avons utilisées pour implémenter les différentes méthodes de segmentation des caractères arabe vus précédemment

4.1.2.1. OpenCv 3.0.0

OpenCV (Open Computer Vision) est une bibliothèque proposant un ensemble de plus de 2500 algorithmes de vision par ordinateur, accessibles au travers d'API pour les langages C, C++, et Python. Elle est distribuée sous une licence BSD (libre) pour les plateformes Windows, GNU/Linux, Android et MacOS.

Initialement écrite en langage C par des chercheurs de la société Intel, OpenCV est aujourd'hui développée, maintenue, documentée et utilisée par une communauté de plus de 40 000 membres actifs. C'est la bibliothèque de référence pour la vision par ordinateur, aussi bien dans le monde de la recherche que celui de l'industrie. Cette bibliothèque nous a aidé à lire et écrire les images (cv2.imread, cv2.imwrite) et les binariser (cv2.threshold), d'appliquer les différentes opérations morphologiques (cv2.dilate, cv2.erode), de détecter le contour d'une image (cv2.Canny), et d'extraire tous les contours présents dans une image (cv2.findContours). [08]**.**

4.1.2.2. NumPy 1.11.3

NumPy [09] est une extension du langage de programmation Python, destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux.

Plus précisément, cette bibliothèque logicielle open source fournit de multiples fonctions permettant notamment de créer directement un tableau depuis un fichier ou au contraire de sauvegarder un tableau dans un fichier, et manipuler des vecteurs, matrices et polynômes.

NumPy est la base de SciPy, regroupement de bibliothèques Python autour du calcul scientifique.

4.1.2.3. SciPy 0.19.0

SciPy [10] est une distribution de modules destinée à être utilisée avec le langage interprété Python afin de créer un environnement de travail scientifique très similaire à celui offert par Scilab, GNU Octave, Matlab… Cette distribution offre plusieurs versions, le choix d'utiliser une version dépend des versions de : noyau de la machine, version de python, version de NumPy.
Elle contient par exemple des modules pour l'optimisation, l'algèbre linéaire, les statistiques, le traitement du signal ou encore le traitement d'images. Il offre également des possibilités avancées de visualisation grâce au module matplotlib.

4.1.2.4. Matplotlib 2.0.0

Matplotlib [11] est une bibliothèque du langage de programmation python qui, combinée avec les bibliothèques python de calcul scientifique *NumPy* et *SciPy*, constitue un puissant outil pour tracer et visualiser des données.

La bibliothèque matplotlib présente de nombreux avantages :

- disponible gratuitement
- open source
- facilité d'apprentissage
- extensible
- marche sur plusieurs systèmes d'exploitation : Unix, Mac Os, Windows, etc.
- une communauté d'utilisateurs de plus en plus importante
- etc.

4.1.2.5. PIL (Python Image Library) 1.1.7

PIL est une bibliothèque pour la manipulation des images. Vous retrouverez de nombreux modules qui vous permettront de traiter des images de n'importe quel format. Cette bibliothèque permet également aux interfaces graphiques construites avec Tkinter [12] d'utiliser des images déclarées avec PIL. [13]

4.2. Construction d'un corpus pour la segmentation du texte arabe

4.2.1. Problématique :

D'après les articles que nous avons lus, nous avons remarqué que les auteurs créent des corpus de test d'une façon aléatoire (numérisation d'un livre, d'un magazine …) et ils donnent des résultats en appliquant leurs méthodes sur leurs propres corpus, ce qui ne permet pas de comparer les résultats des différentes méthodes, puisque chaque méthode est testée sur un

corpus différent. Pour remédier à ce problème il est d'une nécessité importante de créer un corpus standard et unifié.

4.2.2. Les problèmes de la segmentation :

Presque tous les systèmes de segmentation de texte arabe souffrent des problèmes suivants :

1- Le chevauchement des caractères
2- Faux prétraitements :
    a. Extraction de contour fermé ou de squelette.
    b. Détermination de :
        i. Taille du stylo.
        ii. la ligne de base.
        iii. les lignes supérieure et inférieure.
3- Sur-segmentation des caractères ayant plusieurs pics (س، ش، ص، ض، ب، ت، ث، ن).
4- Texte incliné.
5- Espacement condensé entre les caractères du même mot.

4.2.3. Caractéristiques du corpus

Pour garantir que le corpus puisse être utilisé comme référence de tous les systèmes de segmentation, nous avons tenu compte du fait qu'elle doit contenir :

- Toutes les lettres de l'alphabet.
- Les lettres dans les quatre positions : début, fin, milieu et isolée.
- Des caractères en chevauchement.
- Des caractères en ligature verticale.
- Différentes fontes.
- Différentes tailles de police.
- Une qualité d'image moyenne (taille moyenne d'image, bruit...).
- Au moins 4000 caractères.
- Tous les problèmes cités dans la partie ci-dessus.

Pour évaluer notre système de segmentation de texte arabe en caractères, nous avons construit une base de données à partir de **quatre** documents de texte arabe imprimé, ces documents ont été numérisés par un scanner avec une résolution de **700ppp.** La base de données se compose de **71 lignes**, **1330 mots** arabes de **différentes tailles**, contenant au total **4892 caractères** écrits avec les **27 fontes** suivants :

- Courier New
- AL-Sayf Bold
- A Mosalas
- Yakout Linotype Light
- ACS Almass
- ACS Morgan

- AF_Buryidah
- AGA Nada Regular
- AL-Battar
- AF_Jeddah
- FS_Cairo
- Al-Hadith2
- DecoType Naskh Extensions
- FS_Ahram
- MCS Jeddah S_U normal.
- Hesham Free
- AdvertisingLight
- MCS Masahif S_U normal.
- Times New Roman (En-têtes)
- SC_AMEEN
- Arial (Corps)
- Traffic
- Segoe UI Semilight
- arabswell_2
- Mohammad Laha
- khalaad al-arabeh
- AF-Al Nassrah

La figure 61 montre des exemples de texte du corpus construit.


عدة أسابيع قضى خلالها أعضاء نادي الأمل بكلية العلوم والتقنيات بفاس أياماً بلياليها شهدت على العمل الدؤوب والتخطيط وجمع التبرعات تحضيراً لحملة الأمل الخيرية في دورتها التاسعة .

الاستنارة برأيهما. وأخذ مشورتهما في أمور الحياة إذا ما ـ الإنسان انقطع عمله إلا من ثلاثة إلا من صدقة جارية أو علم يُنتفع به أو ولد صالح يدعو له ". رواه مسلم والترمذي. قضاء

تقضينه؟ قالت: نعم. قال: فدين الله أحق أن يقضى ". متفق عليه. تنفيذ ما يوصيان به إن كان لهما وصية. وهي الثلث فأقل: وتنفيذ الوصية واجب. قضاء ما عليهما من صيام الفرض في

النّيابة فيه. قضاء ما عليهما من الطّلّاحات، مثل طلّاحة اليمين، وطلّاحة القتل الخطأ، وغيرها، لأنّ هذه الواجبات تدخل في قوله في حديث ابن

مساعدتهما في أعمالهما، فليس من حسن الخلق أن يرى الولد أباه أو أمّه يعملان ويقف ليتفرّج عليهما. عدم إزعاجهما في نومهما، أو إحداث الجلبة والصّوت الشّديد .

Figure 61 : exemple de texte arabe utilisé dans le corpus

4.3. Résultats

4.3.1. Segmentation du texte en lignes

Dans cette première étape de segmentation du texte en plusieurs lignes, le document contenant le texte arabe a été numérisé et convertit en niveau de gris. L'image résultante est alimentée au système de segmentation. La Figure 62 montre une image d'entrée contenant 10 lignes de texte, écrites avec 6 fontes différentes.


تقديم حقّ الأمّ، وتقديرها، والعطف عليها، والإحسان لها، وذلك لما جاء في حديث
أبي هريرة رضي الله عنه قال: "جاء رجل إلى النبي – صلى الله عليه وسلم – فقال: يا رسول الله من أولى النّاس
بحسن صحابتي؟ قال: أمك، قال: ثم من؟ قال: أمك، قال: ثم من؟ قال: أمك، قال: ثم من؟ قال: أبوك "، رواه البخاري.
معنى البرّ ما كان ضدّ العقوق، وقال ابن الأثير: "البِرّ بالكسر الإحسان، ومنه الحديث في برّ الوالدين، وهو في
حقّهما وحقّ الأقربين من الأهل ضدّ العقوق: وهو الإساءة إليهم والتضييع لحقوقهم "، وبرّ الوالدين بشكل
الإحسان إليهما بالقلب، والقول، والفعل تقرّباً إلى الله تعالى. ويقابله عقوق الوالدين، ويعني إغضابهما من
خلال ترك
الإحسان إليهما، وقيل أنّ عقوق الوالدين هو كلّ فعل يتأذّى منه الوالدان، مع كونه ليس من الأفعال الواجبة.
من أشكال البرّ بالوالدين في حياتهما، ما يلي: طاعة الوالدين والابتعاد عن معصيتهما،
وتقديم طاعتهما على طاعة كلّ البشر، إن لم يكن في ذلك معصية لله عزّ وجلّ أو رسوله

Figure 62 : image d'entrée contenant un texte arabe

Cette image est binarisé, puis elle subit une transformation morphologique « ouverture » pour l'élimination du bruit. Le résultat de cette opération est illustré dans la figure 63.


تقديم حقّ الأمّ، وتقديرها، والعطف عليها، والإحسان لها، وذلك لما جاء في حديث
أبي هريرة رضي الله عنه قال: "جاء رجل إلى النبي – صلى الله عليه وسلم – فقال: يا رسول الله من أولى النّاس
بحسن صحابتي؟ قال: أمك، قال: ثم من؟ قال: أمك، قال: ثم من؟ قال: أمك، قال: ثم من؟ قال: أبوك "، رواه البخاري.
معنى البرّ ما كان ضدّ العقوق، وقال ابن الأثير: "البِرّ بالكسر الإحسان، ومنه الحديث في برّ الوالدين، وهو في
حقّهما وحقّ الأقربين من الأهل ضدّ العقوق: وهو الإساءة إليهم والتضييع لحقوقهم "، وبرّ الوالدين بشكل
الإحسان إليهما بالقلب، والقول، والفعل تقرّباً إلى الله تعالى. ويقابله عقوق الوالدين، ويعني إغضابهما من
خلال ترك
الإحسان إليهما، وقيل أنّ عقوق الوالدين هو كلّ فعل يتأذّى منه الوالدان، مع كونه ليس من الأفعال الواجبة.
من أشكال البرّ بالوالدين في حياتهما، ما يلي: طاعة الوالدين والابتعاد عن معصيتهما،
وتقديم طاعتهما على طاعة كلّ البشر، إن لم يكن في ذلك معصية لله عزّ وجلّ أو رسوله

Figure 63: résultat de prétraitement

Puis cette image binaire est segmentée en plusieurs lignes en appliquant la projection horizontale. Notre système de segmentation fournit une image pour chaque ligne comme le montre la Figure 64.

Figure 64 : résultat de la segmentation de texte en lignes

Le taux de réussite de cette étape est 100%.

4.3.2. Segmentation des lignes en pseudo-mots

Dans cette étape, chaque ligne (image) résultante de l'étape précédente, est segmentée en plusieurs pseudo-mots, en utilisant la projection verticale, le résultat est sous forme de plusieurs images, chacune contient un pseudo-mot, Figure 65.

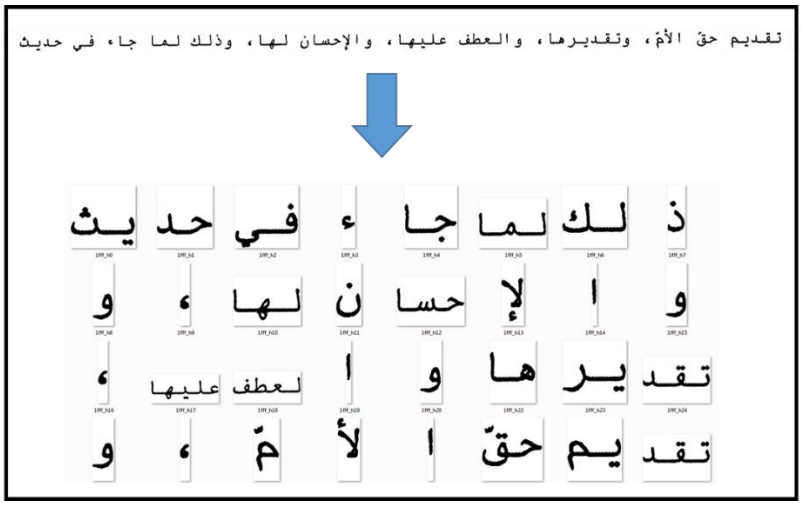


Figure 65 : résultat de la segmentation d'une ligne en plusieurs pseudo-mots

Le taux de réussite de cette étape est 100%.

4.3.3. Segmentation des pseudo-mots en caractères

La segmentation des pseudo-mots en caractère prend en entrée dans chaque itération, une image parmi les images des pseudo-mots résultants de l'étape précédente.

4.3.3.1. Template-matching

Pour cette méthode, nous avons proposé des améliorations dans (Chapitre 3.D.1). Ces améliorations ont permis d'augmenter le taux de réussite de **79.86% à 83.45**%. La figure 66 montre quelques résultats de nos améliorations.

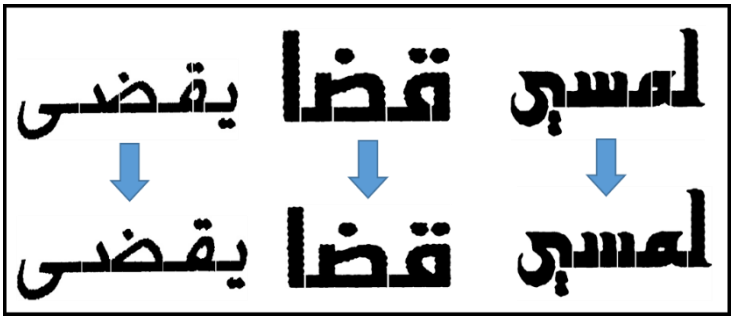


Figure 66 résultats des améliorations proposées pour Template-Matching

La figure 67 donne des exemples d'échec de segmentation.

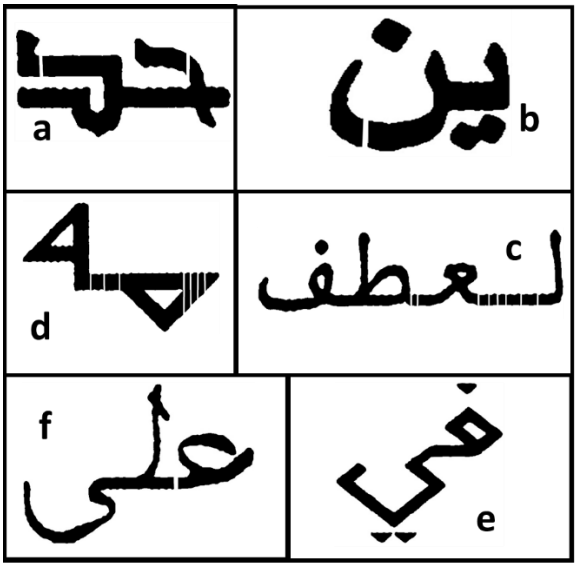


Figure 67 : échec de segmentation pour Template-Matching

Les pseudo-mots en (**a** et **b**) ne sont pas correctement segmentés à cause d'une fausse estimation de la ligne de base, alors que dans (**c** et **d**), la sur-segmentation est une conséquence de la mauvaise qualité de numérisation du document ou de binarisation.

Avec un taux d'erreur de **20.14%,** cette méthode –Template Matching – n'est pas très prometteuse pour la segmentation des caractères arabes. Elle est très sensible au bruit et à la qualité de numérisation du document. Elle est également incapable de segmenter les caractères descendants tels que les derniers caractères de (**e** et **f**) Figure 66.

4.3.3.2. Méthode hybride : contour et Template-matching

La segmentation des pseudo-mots en caractères en utilisant l'analyse du contour et Template-matching a pu donner des meilleurs résultats, car elle profite des avantages de contour et ceux de Template-matching pour localiser les points de segmentation. La Figure 69 montre quelques exemples de segmentation.

Figure 68 : résultat de segmentation

L'opération de segmentation jusqu'à présent consiste à créer le vide entre les caractères dans les points de segmentation, ensuite les caractères sont extraits et sauvegardés dans des images séparées. L'extraction des caractères individuels se fait en traçant les contours du pseudo-mot segmenté, chaque contour fermé appartenant à la zone médiane représente un caractère. Comme montre la figure 70 suivante.

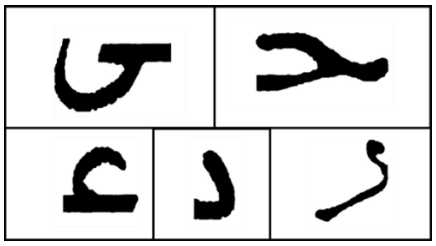
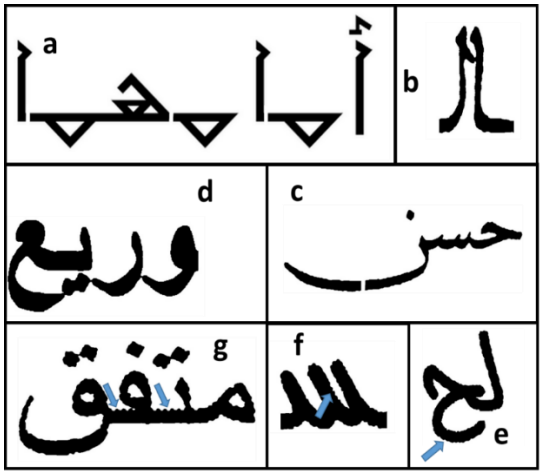


Figure 69 : extraction des caractères

Le taux de réussite de segmentation des pseudo-mots en caractères, en utilisant la méthode hybride proposée, est de **94.76**%. La figure 70 montre des cas d'échec de segmentation.



a : les caractères (ـمـ) connectés sous la ligne de base ne peuvent pas être segmentés, car, pour le contour, il n'y a pas de minimum local entre (ـهـ et ـمـ).

b, d : A cause de la mauvaise qualité de numérisation ou de binarisation, les caractères peuvent se toucher au-dessus ou au-dessous de la ligne de base, et il se voient comme un seul caractère pour la machine.

c : une mauvaise estimation de la ligne de base peut donner également une fausse segmentation. Ici le caractère (ن) est considéré la ligne de base pour le pseudo-mot.

g, f, e : La mauvaise qualité de numérisation peut donner naissance au bruit qui implique la coupure du contour, surtout dans les angles. Cela donne un contour ouvert du pseudo-mot, et par conséquence, la segmentation devient impossible.

4.3.4. Comparaison des résultats

Dans le chapitre 2, nous avons étudié et implémenté des méthodes de segmentation du texte arabe imprimé en caractères. Une comparaison des résultats avec notre hybride méthode est donnée par le tableau 4 :

méthode	caractères bien segmentés	Taux de réussite %
Template-Matching	3907	**83.45**
Projection à base de seuil	3799	77.25
Projection Verticale	3876	79.23
Méthode proposé	**4633**	**94.7**

Tableau 4 : Comparaison des méthodes de segmentation

Conclusion

Dans ce travail, nous avons proposé un système de segmentation des textes arabe en caractères individuels écrits en **multi-fonte** et **multi-taille.** Nous avons essayé au maximum d'implémenter nos propres idées dans les diverses étapes de segmentation. Nous avons pu augmenter le taux de réussite de **86.19%** à **94.76%.** La majorité d'effort était consacré à la méthode basée sur le contour, car elle a donné les meilleurs résultats pour notre base de données (Chapitre 2.E). Ces résultats sont très prometteurs, et ils peuvent être améliorés dans la phase de reconnaissance. Le problème que nous avons rencontré dans ce travail, est la grande taille des échantillons de la base de données, ce qui influence négativement sur le temps de réponse du système. Car, pour avoir une bonne résolution et des clairs détails, les images utilisées doivent être de grande taille, ce qui demande plus de temps pour leur traitement.

Conclusion et perspectives

Malgré les efforts et les travaux intensifs réalisés dans le domaine de la reconnaissance optique de l'écriture, aucun système OCR n'est jugé fiable à 100% et le problème de segmentation n'est pas encore résolu. Mais au fur et à mesure les auteurs essayent d'améliorer les scores pour de meilleurs résultats.
Les problèmes majeurs influençant la recherche en OCR arabe sont, l'absence d'outils tels que dictionnaires, bases de données et statistiques se rapportant à l'écriture arabe.

Pendant ce projet de fin d'étude, nous avons pu explorer le domaine de reconnaissance optique de caractère pour le texte arabe imprimé. Nous avons découvert les différentes approches utilisées et nous avons vu et analysé plusieurs travaux réalisés dans le domaine de segmentation de texte arabe imprimé. Et pour pouvoir comparer les différentes méthodes de segmentation que nous avons implémentées, nous avons construit un corpus unifié de 4892 caractères, écrits avec 27 fonts différentes. Ce corpus est construit en tenant compte tous les cas qui peuvent poser des difficultés pour un système de segmentation. Dans notre principale contribution, nous nous sommes intéressés à la phase la plus délicate dans un système OCR arabe, celle de segmentation du texte en caractères, qui reste un domaine vaste de recherche. La segmentation du texte arabe présente plusieurs difficultés. C'est la phase où se produisent la plupart des erreurs et où l'erreur de segmentation entraînera des erreurs de classification par la suite. Nous avons construit une méthode hybride de segmentation du texte arabe en caractères en se basant sur l'approche du contour et celle de la Template-Matching. Cette nouvelle méthode est étudiée et développée de manière à ce que la segmentation soit effectuée en minimisant les erreurs et en maximisant le taux de réussite pour le texte arabe imprimé en multi-fonte et multi-taille.

Nous avons obtenu des résultats très satisfaisants (**94.76%**), que nous essayerons encore de les améliorer sous l'encadrement des professeurs du LSIA FSTF, espérant pouvoir contribuer à la recherche dans le domaine de l'OCR arabe.

Sous l'encadrement de nos professeurs de laboratoire LSIA, nous avons pu soumettre un papier scientifique « **A new hybrid method for printed Arabic text segmentation, and a reference corpus construction** » [LAAI17] à la 6ème Conférence internationale sur le traitement du langage arabe ICALP'17 en présentant notre travail dans ce projet.

Perspectives :

Ce travail peut être la base d'un système de reconnaissance de texte arabe et d'autres langues. Aussi comme perspectives, c'est l'amélioration de la méthode de segmentation, le travail sur d'autres approches tel que la squelettisation, le développement des systèmes de segmentation dynamiques en utilisant Machine Learning, et l'adaptation de la méthode de segmentation pour traiter les textes manuscrits qu'ils soient arabes ou d'autres langues où le texte est cursif.

Références

[AHT04] Zheng, L., Hassin, A.H., Tang, X.: A new algorithm for machine printed Arabic character segmentation. Patt. Recognit. Lett. 25(15), 1723–1729 (2004)

[AM82] Adnan Amin and Masini G., Machine Recognition of Arabic Cursive Words, SPIE 26th International Symposioum on Instrument Display, Application of Digital Image Processing IV, Vol. 359,San Diego, pp. 286-292, Aug 1982.

[AM89] Amin, A.,Mari, J.:Machine recognition and correction of printed Arabic text. IEEE Trans. Syst. Man Cybern. SMC 19(5), 1300–1306 (1989)

[AS91] A. Amin and S. Al-Fedaghi, Machine Recognition of Printed Arabic Text Utilizing a Natural Language Morphology, International Joumal of Man-Machine Studies IJMMS, 35(6), pp. 769-788, 1991

[BM81] B. Parhami and M. Taraghi, Automatic Recognition of Printed Farsi Texts, Pattern recognition, 14(1-6), pp. 395-403, 1981.

[BS97'] Bushofa, B., Spann, M.: Segmentation and recognition of Arabic characters by structural classification. Image Vis. Comput.(IVC) 15(3), 167–179 (1997).

[BS97] Bushofa, B.M.F., Spann, M. Segmentation of Arabic characters using their contour information. 13th International Conference on Digital Signal Processing, vol. 2, pp. 683–686. 1997

[BRG14] Bharatratna P. Gaikwad Ramesh R. Manza Ganesh Manza :Automatic Video Scene Segmentation to Separate Script for OCR. International Journal of Computer Applications (0975 – 8887) Recent Advances In Information Technology-2013-14

[CHKY96] C. Olivier, H. Miled, K. Romeo-Pakker and Y. Lecourtier, Segmentation and Coding of Arabic Handwritten Words, International Conference on Pattern Recognition (ICPR '96), Vienna, Austria, Vol. 3, pp. 264-268, 25-29 Aug 1996.

[ES90] El-Khaly, F., Sid-Ahmed,M.A.:Machine recognition of optically captured machine printed Arabic text. In: Proceedings of Pattern Recognition, vol. 23, pp.1207–1214 (1990)

[JF01] J. Cowell and F. Hussain, Thinning Arabic Characters for feature Extraction. IEEE Conference on Information Visualization, London, pp. 181 -185, 25-27 Jul, 2001.

[GUA92] Goraine, H., Usher, M., Al-Emami, S.: Off-line Arabic character recognition. Computer 25(7), 71–74 (1992)

[HA95] H. AJ-Sadoun and A. Amin, A New Structural Technique for Recognizing Printed Arabic Text, International Joumal of Pattern Recognition and Artificial Intelligence, 9(1), pp. 101-125, 1995.

[H68] H. Freeman, On the Encoding of Arbitrary Geometric Configuration, IEEE Transactions on Electronic Computing EC-1O, pp. 260-268, 1968

[I03] I. Abuhaiba, Arabic Fonte Recognition Based on Templates, The Intemational Arab Journal of IT (IAJIT), 1(0), pp. 33-39, Jul 2003.

[L14] LASRI Younes : Contribution a la reconnaissance optique (OCR) du texte arabe imprimé. Université sidi mohamed ben abdellah, laboratoire LSIA FSTF.

[LAAI17] Y. Lasri, A. Zoizou, A. Zarghili, I. Chaker. "A new hybrid method for printed Arabic text segmentation, and a reference corpus construction". Soumis à "The 6th International Conference on Arabic Language Processing ICALP 2017, October 11th 12th, October 2017."

[LCXH06] Liangrui, P., Changsong, L., Xiaoqing, D., Hua,W.: Multilingual document recognition research and its application in China. Second International Conference on Document Image Analysis for Libraries, pp. 126–132 (2006).

[M92] Margner, V.: SARAT-a system for the recognition of Arabic printed text. 11th International Conference on Pattern RecognitionMethodology and Systems, vol. 2, Conference B, pp. 561–564 (1992)

[NSZA03] Nawaz, S.N., Sarfraz, M., Zidouri, A., Al-Khatib, W.G.: An approach to off-line Arabic character recognition using neural networks. In: Proceedings of the 10th IEEE International Conference on Electronics, Circuits and Systems (ICECS 2003), vol. 3,pp. 1328–1331 (2003)

[S88] S. AI-Emami, Machine Recognition of Handwritten and Typewritten Arabic Characters, PhD thesis, Univ. of Reading, Dept. of Cybernetics, Sept. 1988

[SNA03] Sarfraz, M., Nawaz, S.N., Al-Khuraidly, A.: Off-line Arabic text recognition system. International Conference on Geometric Modeling and Graphics, London, England, pp. 30–36 (2003).

[SS02] Sari, T., Souici, S.M.: Off-line handwritten Arabic character segmentation algorithm: ACSA. In: Proceedings of International Workshop Frontiers in Handwriting Recognition, pp. 452–457 (2002)

[TC84] T.Y.Zhang, C.Y.Suen, A fast parallel algorithm for thinning digital patterns, Communications of the ACM, Volume 27 Issue 3, March 1984, p236-239

[01] Cours en ligne, Hilditch's Algorithm for Skeletonization, Université de McGill, Montréal – Canada.
http://cgm.cs.mcgill.ca/~godfried/teaching/projects97/azar/skeleton.html

[02] Site official de OpenCV
http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/canny_detector/canny_ detector.html

[03] Site Officiel de OpenCV
http://docs.opencv.org/trunk/d7/d4d/tutorial_py_thresholding.html

[04] Site de l'école nationale supérieure de techniques Avancées de Paris, http://perso.ensta-paristech.fr/~manzaner/Cours/IAD/TERI_MorphoMath.pdf.

[05] Wikipédia, https://fr.wikipedia.org/wiki/D%C3%A9tection_de_contours

[06] Site de laboratoire mathématiques et interactions, université de Nice
http://math.unice.fr/~diener/MISM/COURS2.pdf. Consulté le 24 mai 2017.

[07] Page Wikipédia https://fr.wikipedia.org/wiki/Python_(langage)#Utilisation

[08] Cours en ligne, site de zéros. https://openclassrooms.com/courses/introduction-a-la-vision-par-ordinateur.

[09] Site official de NumPy : http://www.numpy.org.

[10] Site official SciPy,
https://docs.scipy.org/doc/scipy/reference/tutorial/general.html.

[11] Site official de Matplotlib : http://matplotlib.org/index.html.

[12] Site official de Tkinter : http://tkinter.fdex.eu/doc/intro.html.

[13] Site Officiel de PIL : https://pillow.readthedocs.io/en/4.1.x/.