N° d'ordre : 135 / 2017

UNIVERSITE SULTAN MOULAY SLIMANE

Faculté des Sciences et Techniques

Béni-Mellal

Centre d'Études Doctorales : Sciences et Techniques
Formation Doctorale : Mathématiques et Physique Appliquées

THÈSE

Présentée par

Youssef OUADID

Pour l'obtention du grade de

DOCTEUR

Spécialité : Informatique

Reconnaissance des Caractères Tifinaghs par les Graphes

Soutenue le 28/10/2017, devant la commission d'examen composée de :

Président	:	Pr. Mustapha AIT LAFKIH	PES	FST - Béni Mellal
Rapporteurs	:	Pr. Mohamed OUKESSOU	PES	FST - Béni Mellal
		Pr. Mostafa JOURHMANE	PES	FST - Béni Mellal
Examinateur	:	Pr. Rachid EL AYACHI	PH	FST - Béni Mellal
Directeur de thèse	:	Pr. Brahim MINAOUI	PES	FST - Béni Mellal
Co-directeur	:	Pr. Mohamed FAKIR	PES	FST - Béni Mellal

© OUADID, 2017

À mes parents qui se sont beaucoup sacrifiés pour mes études

Cette thèse a été réalisée au sein du Laboratoire de Traitement de l'Information et Aide à la Décision (**TIAD**) de la Faculté des Sciences et Techniques Beni Mellal (**FSTBM**) sous la direction du professeur Brahim MINAOUI et l'encadrement du professeur Mohamed FAKIR.

Remerciements

Ce travail a été effectué au sein du Laboratoire de Traitement de l'Information et Aide à la Décision (**TIAD**) de la Faculté des Sciences et Techniques Beni Mellal (**FSTBM**) sous la direction du professeur Brahim MINAOUI. Je tiens à lui écrouler ici toute ma reconnaissance pour sa disponibilité et ses précieux conseils.

J'assure très sincèrement de ma gratitude, le professeur Mohamed FAKIR, qui a dirigé une partie de mon travail et m'a longuement encouragé. Je suis sensible à l'honneur qu'il me fait de participer au jury.

Monsieur le Professeur Mustapha AIT LAFKIH d'avoir accepté d'être président du jury. Je lui adresse mes sincères remerciements.

Je remercie les Professeurs Mostafa JOURHMANE, Mohamed OUKESSOU et Benaceur OUTTAJ pour avoir accepté de rapporter ma thèse, et pour leurs conseils et leurs remarques pertinentes.

Je remercie également le Professeur Rachid EL AYACHI pour l'intérêt qu'il a porté à mon travail et d'avoir accepté de participer au jury en tant qu'examinateur.

Je remercie tous les membres du laboratoire de Traitement de l'Information et Aide à la Décision.

Merci beaucoup à tous mes amis qui, de près ou de loin m'ont aidé et encouragé aux moments opportuns. Je les remercie pour tout le temps précieux que nous avons passé ensemble.

Résumé

Le travail présenté dans cette thèse s'inscrit dans le cadre de la reconnaissance automatique des caractères. Il s'agit de la reconnaissance des caractères Tifinaghs par les graphes.

Dans la première partie de ce travail, nous avons élaboré un système de reconnaissance des caractères Tifinaghs imprimés par appariement exact des graphes décrits par des matrices d'incidence. Les graphes sont construits à l'aide des points clés extraits à partir des squelettes des différents caractères par la méthode des voisinages. Dans un deuxième temps, afin d'améliorer d'avantage le taux de reconnaissance de ce système, nous y avons intégré une méthode pour lisser les squelettes et la méthode de Harris pour extraire les points clés. Au niveau de la classification, nous avons appliqué l'appariement inexact des graphes par une approche spectrale permettant de représenter et de distinguer les propriétés structurelles des graphes à l'aide des vecteurs et valeurs propres de leurs matrices d'adjacence. L'évaluation du système ainsi réalisé, à l'aide d'une base de données des caractères imprimés, a mis en évidence les bonnes performances de ce système en termes de taux de reconnaissance et du temps d'exécution.

Dans la deuxième partie de ce travail, nous nous sommes intéressés à l'élaboration d'un système de reconnaissance des caractères Tifinaghs manuscrits. Vu la complexité et la diversité de la forme manuscrite de ces caractères, nous avons développé des méthodes de traitement plus avancées pour les décrire et les classer, telles que la méthode d'extraction des points clés suffisants pour une représentation graphique compacte et rapide des caractères , et la méthode de classification hybride combinant l'appariement exact des graphes et le réseau bayésien naïf. La mise en œuvre du système intégrant ces méthodes, a mis en évidence que ce système est plus performant que les systèmes existants, en termes de précision et de rapidité.

Mots-clés

Reconnaissance des caractères Tifinaghs ; Théorie des graphes ; Représentation graphique ; Matrice d'incidence ; Matrice d'adjacence ; Approche spectrale ; Appariement des graphes ; Réseau bayésien naïf.

Abstract

The presented thesis deals with the recognition of printed and handwritten Tifinagh characters using a structural representation which is the graphs.

In the first part of this work, we have developed a system for the recognition of printed Tifinagh characters by exact graph matching described by incidence matrices. Graphs are constructed using the key points, extracted from the skeletons of the different characters, by the neighborhood method. Then, in order to improve the recognition rate of this system, we have integrated a procedure to smooth the skeletons and the Harris method to extract the key points. At the classification level, we applied the inexact graph matching by a spectral approach that represent and distinguish the structural properties of graphs using the eigenvectors and eigenvalues of their adjacency matrices. The evaluation of the system thus carried out, using a database of printed characters, highlighted the good performances of this system in terms of recognition rate and execution time.

In the second part of this work, we were interested in the development of a system for the recognition of handwritten Tifinagh characters. Given the complexity and diversity of the handwritten form of these characters, we have developed more advanced processing methods to describe and classify them, such as a sufficient key points extracting algorithm for a compact and fast graphical representation of the characters, And a hybrid classification method combining the exact matching of the graphs and the naive Bayesian network. The implementation of the system integrating these methods has shown that this system is more efficient than the existing systems in terms of accuracy and speed.

Keywords

Graph theory; Graph matching; Graph representation; Structural feature extraction; Harris method; Recognition of isolated Tifinagh characters; Incidence matrix; Adjacency matrix.

ملخص

العمل المقدم في هذه الرسالة يهم التعرف التلقائي على حروف اللغة الامازيغية (تيفيناغ). وذلك بالاعتماد على مخططات الحروف.

في الجزء الأول من هذا العمل قمنا بتطوير نظام للتعرف على الحروف المطبوعة يعتمد على التطابق الدقيق لمخططات هذه الأخيرة التي تمثلها مصفوفات الارتباط. هده المخططات يتم إنشاؤها من خلال نقط رئيسية يتم استخراجها من خلال هياكل الحروف باستعمال طريقة الجوار. من جهة ثانية، وبغرض تحسين نسبة التعرف لهذا النظام، قمنا بصقل هياكل الحروف وباستخدام طريقة هاريس لاستخراج النقاط الأساسية. فيما يخص طريقة التصنيف، استعملنا التطابق الغير الدقيق للمخططات عن طريق مقاربة طيفية تمكن من تقديم وتمييز الخصائص البنيوية لهده المخططات من خلال القيم والمتجهات الذاتية المستخرجة من مصفوفات الجوار. تم تقييم النظام المنجز باستخدام قاعدة بيانات لحروف مطبوعة. هذا التقييم برهن على كفاءة النظام من جهة نسبة التعرف وكذلك سرعة وقت التنفيذ.

في الشطر الثاني من هذا العمل، صببنا اهتمامنا على تطوير نظام للتعرف على الحروف المكتوبة. نظرا لصعوبة وتنوع أشكال الحروف المكتوبة يدويا، قمنا بخلق طرق أكثر تطورا لمعالجة هذا النوع من الحروف لوصفها وتصنيفها، كطريقة استخراج النقط الرئيسية كافية لتمثيل المخططات بشكل مختزل وسريع للحروف، وطريقة التصنيف الهجينة التي تجمع بين التطابق الدقيق للمخططات والشبكة البايزية. هذا النظام المنجز برهن على كفاءته بالمقارنة مع الأنظمة الموجودة فيما يخص الدقة والسرعة.

الكلمات الدالة

التعرف على حروف اللغة الأمازيغية؛ المخططات؛ تمثيل المخططات؛ مصفوفات الارتباط؛ مصفوفات الجوار؛ المقاربة الطيفية؛ تطابق المخططات، الشبكات البايزية.

Table de matières

Table des figures

Liste des tableaux

Liste des abréviations

OCR	**Reconnaissance Optique des Caractères**
SVM	**Machine à Support de Vecteurs**
K-NN	**K Plus Proches Voisins**
PSC	**Problèmes de Satisfaction de Contraintes**
AGR	**Graphes Relationnels Attribués**
DFG	**Distance des Graphes Fonctionnels**
EM	**Relaxation de l'étiquetage et la maximisation d'attente**
NN	**Plus Proches Voisins**
LVQ	**Quantification de Vecteurs d'Apprentissage**
SOM	**Cartes Auto-Organisatrices**
MLP	**Perceptron Multicouche**
PCA	**Analyse des Composants Principaux**
MDS	**Échelle Multidimensionnelle**
IRCAM	**Institue Royale de la Culture Amazighe**
RNA	**Réseaux de Neurones Artificiels**
HMM	**Modèle de Markov Caché**

1 Introduction générale

1.1 Contexte et motivation

Depuis des millénaires, l'être humain a toujours inventé des techniques visant sa pérennité à travers les générations. Parmi ces inventions pionnières, l'écriture a été utilisée comme moyen de communication pour représenter le langage et l'émotion, et pour conserver et archiver le savoir. L'écriture a été le précurseur pour inventer plusieurs technologies à travers les générations. En effet, avec l'apparition de l'imprimerie, la quantité d'informations archivées sur des papiers a connu une augmentation exponentielle. En raison de leurs fragilités vis-à-vis des facteurs naturels et de l'apparition de nouveaux techniques (ordinateur, scanner, appareil photo numérique…), il a été évident de penser à développer des outils pour communiquer ces informations à la machine tel que les systèmes de reconnaissance optique des caractères (OCR). Ces systèmes sont des procédés informatiques qui permettent de convertir une image de texte en un texte compréhensible par la machine. Bien que, la mise en œuvre de tels systèmes est extrêmement difficile, vue la grande variabilité liée aux habitudes des scripteurs ainsi qu'aux styles et formes d'écriture, de nombreuses techniques d'OCR ont été améliorées et perfectionnées, notamment pour les écritures Latines et Arabes. Quant à l'écriture Amazighe, en raison de sa récente intégration dans les systèmes d'informations, l'élaboration d'un système de reconnaissance optique performant de cette écriture est actuellement un champ de recherche ouvert.

En effet, la langue Amazighe est la plus ancienne écriture de l'Afrique du Nord, elle a plus de 3000 ans d'existence. Au Maroc, le processus de légitimation des langues «maternelles» et en particulier l'Amazighe a débuté véritablement en 1994 suite au discours Royal du 20 août. Le Roi Hassan II y déclare en effet qu'il convient d'envisager l'introduction des dialectes dans les programmes scolaires. Suite à ses premières directives royales, la Charte Nationale d'Éducation et de Formation, élaborée en octobre 1999 dans le cadre de la réforme de l'enseignement, est validée par le Roi Mohammed VI. Elle intègre, parmi les 19 leviers qui sont des propositions pour le changement, le levier 9 qui est relatif à l'introduction de la langue Amazighe dans l'enseignement. Mais c'est avec le discours Royal d'Ajdir (Khénifra) du 17 octobre 2001 que la légitimation de la langue Amazighe est officialisée, puisqu'elle institue, par un dahir, la création et l'organisation de l'Institut Royal de la Culture Amazighe,

concrétisant par-là l'annonce de sa fondation par le Roi Mohammed VI lors du Discours du Trône du 30 juillet 2001. Cette institution est « chargée de sauvegarder, de promouvoir et de renforcer la place de notre culture Amazighe dans l'espace éducatif, socioculturel et médiatique national ainsi que dans la gestion des affaires locales et régionales (…) » (motif 8 du dahir). Elle a élaboré et adopté l'alphabet Tifinagh comme caractère de l'écriture de la langue Amazighe. Depuis l'introduction de cet alphabet dans le système universel de codage en 2004, des efforts des centres de recherche appartenant à l'IRCAM se sont manifestés dans de nombreuses et différentes études approfondies portant sur la promotion de cet alphabet, l'élargissement de son rayonnement et son intégration dans les systèmes d'informations. Ce qui a entraîné l'apparition des documents en langue Amazighe écrits par les caractères Tifinaghs imprimés ou manuscrits. Dès lors, le traitement automatique de ces documents est devenu un champ de recherche très actif. Le but est de réaliser des systèmes de reconnaissance automatique de texte permettant la transformation du texte de tels documents en une représentation compréhensible par une machine et facilement reproductible par un système informatique. Cependant, les travaux réalisés dans ce contexte se sont heurtés au problème de la confusion des caractères. La recherche d'une solution à cette problématique nous a motivé à lancer des travaux de recherche pour élaborer de nouvelles méthodes de reconnaissance automatique des caractères Tifinaghs.

L'écriture Amazighe (Tifinagh) est composée de 33 graphèmes correspondant aux 33 phonèmes de l'Amazighe standard. Le tableau 1 présente ces 33 caractères ainsi que leurs correspondants en caractères Latins [1].

*Tableau 1-1.*Le répertoire *officiel de l'alphabet Tifinagh*-IRCAM avec leurs correspondants en caractères latins

ya	ⵄ	a	yaḥ	ⵅ	ḥ	yaṛ	ⵕ	ṛ
yab	ⵃ	b	yaɛ	ⵀ	ɛ	yaɣ	ⵖ	ɣ
yag	ⵗ	g	yax	ⵠ	x	yas	ⵙ	s
yagʷ	ⵗⵯ	gʷ	yaq	ⵇ	q	yaṣ	ⵚ	ṣ
yad	ⵏ	d	yi	ⵉ	i	yac	ⵛ	c
yaḍ	ⵎ	ḍ	yaj	ⵊ	j	yat	ⵜ	t
yey	ⵢ	e	yal	ⵍ	l	yaṭ	ⵟ	ṭ
yaf	ⵋ	f	yam	ⵎ	m	yaw	ⵡ	w
yak	ⴽ	k	yan	ⵏ	n	yay	ⵣ	y
yakʷ	ⴽⵯ	kʷ	yu	ⵓ	u	yaz	ⵣ	z
yah	ⵀ	h	yar	ⵔ	r	yaẓ	ⵥ	ẓ

L'écriture Amazighe est écrite de gauche à droite. Contrairement à la langue arabe et latine, elle n'a pas la notion de majuscule et minuscule, ni celle de pseudo-mot. Cependant, elle comprend les mêmes signes de ponctuation que celle de la langue latine. C'est une écriture non cursive ce qui facilite le processus de segmentation. Cela justifie le fait que la plupart des recherches effectués sur cette écriture sont concentrées sur la reconnaissance des caractères vu que les travaux existants sur la segmentation des documents de l'écriture latine sont viable pour celle de l'Amazighe.

Concernant les modèles graphiques des caractères, leurs formes sont composées de points, de petits cercles et de segments droites. En effet, on peut diviser ces caractères en trois classes :

- Lettres linéaires : ⵅ, ⵍ, ⴻ, ⵀ, ⵇ, ⵅⵯ, ⵅ, ⵔ, ⵄ, ⵣ, ⵉ, ⵏ, ⵛ, ⵏ, ⵢ, ⵜ, ⴻ, ⵡ, ⵥ, ⵇⵯ ;
- Lettres circulaires : ⴰ, ⵔ, ⵖ, ⵙ, ⵚ, ⵔ, ⵟ, ⵙ, ⵣ ;
- Autres : ⵋ, ⵅ, ⵯ ;

Si on ignore les critères de l'orientation et de la taille dans les modèles graphiques des caractères, en remarque que certains caractères sont similaires tels que : (ⵔ et ⵙ), (ⵛ et ⵡ), (ⴰ et ⵔ). Il est important de noter que la plupart des caractères sont composés d'un seul composant connexe à part quelques alphabets listés dans la figure 1.1 ci-dessous.

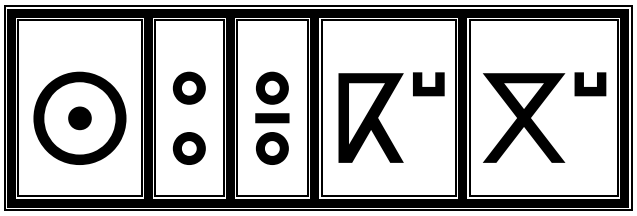


Figure 1-1. Caractères Tifinaghs composés de plusieurs composants connexes

1.2 Difficulté et défi

La principale difficulté dans l'élaboration d'un système d'OCR performant réside dans la confusion entre les caractères. En effet, la reconnaissance automatique des images des caractères consiste à décrire automatiquement le contenu des images par des descripteurs à travers une analyse de leur contenu visuel. Cette analyse est confrontée par le bruit produit l'or de l'acquisition, par le problème de la variabilité de la forme et par les variations qui peuvent parvenir de changement d'échelle, de changements d'inclinaison et de la rotation. Ceci accentue le problème de la variation intra-classes et de la ressemblance interclasses.

Cette variation visuelle du contenu des images des caractères crée des relations compliquées entre le niveau du contenu visuel et le niveau des classes des caractères, ce qui entraine une confusion entre les caractères et rend le problème de reconnaissance très difficile à résoudre en particulier dans le cas des caractères manuscrits.

Dans le cas des caractères Tifinaghs, les travaux réalisés pour résoudre ce problème ont mis en évidence qu'une combinaison adéquate des descripteurs de forme et/ou une hybridation appropriée des classificateurs permet de réduire la confusion et d'augmenter par conséquence le taux de reconnaissance des caractères, mais au détriment du temps de reconnaissance qui devient plus prohibitif. Ainsi, la réduction de la confusion des caractères et leurs temps de reconnaissance constitue le défi majeur des systèmes d'OCR. Afin de relever ce défi, nous avons recouru à une approche structurelle pour la description des caractères Tifinaghs qui réside dans la représentation des caractères par des graphes.

L'objectif de notre travail consiste à proposer une représentation graphique définissant la description structurelle des caractères Tifinaghs et de mettre en œuvre des méthodes de classification de ces représentations.

1.3 Contribution

Dans le présent travail, nous avons élaboré des systèmes de reconnaissances des caractères Tifinaghs imprimés et manuscrits par les graphes. Les résultats de notre travail ont donné lieu aux publication et communications suivantes :

❖ **Publications:**

- Ouadid, Y., Fakir, M., & Minaoui, B. (2016). Tifinagh Printed Character Recognition through Structural Feature Extraction. *International Journal of Computer Vision and Image Processing (IJCVIP), 6*(2), 42-53.
- Ouadid, Y., Boutaounte, M., Fakir, M., & Minaoui, B. (2016). Tifinagh Character Recognition using Harris Corner Detector and Graph Representation. *International Journal of Computer Applications,* 149(3), 17-23.
- Ouadid, Y., Minaoui, B., & Fakir, M. (2016). Spectral Graph Matching for Printed Tifinagh Character. In *Computer Graphics, Imaging and Visualization (CGiV), 2016 13th International Conference on* (pp. 105-111). IEEE.

- Ouadid, Y., Minaoui, B., Fakir, M., Aboulala, O., Boutaounte, M., Elbalaoui, A. (2014). Pattern Recognition by Graphs: Application to Printed Tifinagh Character. *International Journal of Advanced Research in Computer Science & Technology (IJARCST)*, 2(1).

❖ **Communications:**

- Ouadid, Y., M., Elbalaoui, O., Boutaounte, Fakir, M., & Minaoui, B. (2017). Handwritten Tifinagh Character Recognition using graphs. The Third *International Conference on Business Intelligence (CBI'17).* FST Béni Mellal, Morocco.
- Ouadid, Y., Minaoui, B., & Fakir, M. (2016) Reconnaissance des caractères Tifinagh Imprimés par une Description Structurelle. *7ème Edition de la Conférence Internationale sur Les Technologies d'Information et de Communication pour l'Amazighe TICAM'16*. IRCAM, Rabat, Maroc.
- Ouadid, Y., Minaoui, B., & Fakir, M. (2016). Spectral Graph Matching for Printed Tifinagh Character. In *Computer Graphics, Imaging and Visualization (CGiV), 2016 13th International Conference on* (pp. 105-111). IEEE.
- Ouadid, Y., Minaoui, B., Fakir, M. & Aboulala, O. (2015). Spectral Graph Matching and Harris Corner Detector for Printed Tifinagh Character Recognition. The Second *International Conference on Business Intelligence (CBI'15).* FST Béni Mellal, Morocco.
- Ouadid, Y., Minaoui, B., Fakir, M. & Aboulala, O. (2014). Tifinagh Character Recognition by Graphs. *First International Conference on Business Intelligence (CBI'14).* FST Béni Mellal, Morocco.
- Ouadid, Y., Minaoui, B. & Aboulala, O. (2014). Exact Graph Matching using Incidence Matrix : Recognition of Tifinagh Character. *6ème édition des journées doctorales en Technologies de l'Information et de la Communication (JDTIC'14)*. ENSIAS, Rabat, Maroc.
- Ouadid, Y., Minaoui, B., Fakir, M., & Abouelala, O. (2014). New Approach of Tifinagh Character Recognition using Graph Matching. *6ème Edition de la Conférence Internationale sur Les Technologies d'Information et de Communication pour l'Amazighe TICAM'14.*

- Ouadid, Y., Minaoui, B., Fakir, M. & Aboulala, O. (2014). Pattern Recognition by Graphs : Application to Printed Tifinagh Character. *Congrès Méditerranéen des Télécommunications (CMT'14).* FST Mohammedia, Maroc.
- Ouadid, Y., Minaoui, B., Fakir, M. & Aboulala, O. (2014). Pattern Recognition by Graphs. *Première édition des Journées Docotoriales*. Béni Mellal, Morocco.

❖ **Prix :**

- Meilleur présentation orale à la 7ème Conférence Internationale sur les Technologies d'Information et de Communication pour l'Amazighe (TICAM'16), IRCAM, Rabat, Maroc, novembre 2016.

1.4 Organisation du manuscrit

La suite du manuscrit est organisée selon deux parties :

- La première partie présente un état d'art sur la reconnaissance optique des caractères et la reconnaissance des formes par les graphes. Cette partie est constituée par les deux chapitres suivants:
  - **Chapitre 1 :** Ce chapitre décrit le principe, les aspects et l'architecture d'un système de reconnaissance optique des caractères, ainsi que les différentes techniques utilisées dans ce domaine ; puis, présente un état d'art sur la reconnaissance des caractères Tifinaghs.
  - **Chapitre 2 :** Dans ce chapitre un aperçu sur la reconnaissance des formes par les graphes est présenté. Il décrit la façon dont les graphes ont été utilisés dans le domaine de reconnaissance des formes, ainsi que les avantages de la description graphique par rapport à la description statistique.
- la deuxième partie présente notre contribution dans la reconnaissance des caractères Tifinaghs. Cette partie est constituée par deux chapitres :
  - **Chapitre 3 :** Ce chapitre présente les systèmes de reconnaissance des caractères Tifinaghs imprimés par les graphes et décrit les expériences réalisées.
  - **Chapitre 4 :** Ce chapitre est consacré à la présentation des systèmes de reconnaissance des caractères manuscrits et à l'étude de leurs performances.

Dans la conclusion de ce mémoire, nous présentons un bref aperçu sur nos contributions tout en proposant de multiples perspectives de recherche concernant la reconnaissance des caractères Tifinaghs.

PREMIÈRE PARTIE : ÉTUDES BIBLIOGRAPHIQUES

2 Reconnaissance optique des caractères

2.1 Introduction

La reconnaissance optique des caractères (OCR) relève de la reconnaissance des formes et de l'intelligence artificielle. Elle contribue énormément à la communication homme-machine. C'est le processus informatique de conversion d'images de texte imprimé ou manuscrit en un texte compréhensible par la machine et facilement reproductible par un système informatique (Microsoft Word par exemple), que ce soit à partir d'un document scanné ou d'une photo d'un document. L'OCR concerne plus précisément les formes d'information acquises à partir d'un document imprimé en papier, que ce soit des factures, des reçus informatisés, des documents de passeport, des relevés bancaires, des courriers ou toute documentation appropriée. La réalisation d'un tel processus est très compliquée car il existe une infinité de représentations d'écriture. En effet, chaque personne a un style d'écriture unique et il existe de nombreux styles et fonts en caractères imprimés. Donc, les systèmes de reconnaissance des caractères sont adaptés au type de l'écriture envisagé (imprimé, manuscrit ou cursif).

Dans ce chapitre, nous présentons une description générale des différents éléments et aspects concernant le processus de l'OCR.

2.2 Aspects de l'OCR

La réalisation d'un OCR dépend énormément du type de données traitées ainsi de l'application visée. Selon ces contraintes, Al Falou et al. [2] ont donnés cinq aspects principaux de l'OCR qu'on va définir dans la suite de cette section.

2.2.1 Reconnaissance de l'imprimé ou du manuscrit

L'approche change selon le type de l'écriture. Dans le cas de l'imprimé, les caractères sont souvent non cursifs et bien alignés, ce qui simplifie la phase d'acquisition. De plus, la fonte constitue un modèle idéal pour l'identification. Par contre, dans le cas du manuscrit, les caractères sont souvent mal inclinés et leur fonte est indéterminée ce qui nécessite plus de processus comparées aux imprimés.

2.2.2 Reconnaissance mono fonte, multi fonte ou omni fonte

Lorsqu'il s'agit des fontes déterminées, ces notions ne concernent que les systèmes de reconnaissance imprimée. Un système de reconnaissance mono fonte ne traite qu'une seule fonte d'écriture, ce qui simplifie la phase d'apprentissage. Un système est dit multi fonte, lorsque plusieurs fontes sont traitées, dans ce cas un prétraitement est nécessaire pour réduire l'écart entre les caractères (taille et épaisseur par exemple). Dans le cas des systèmes omni fonte, la fonte n'est pas importante puisque le système est supposé reconnaitre n'importe quelle fonte sans l'avoir apprise, ce qui a nécessité une nouvelle approche dans le domaine de la recherche.

2.2.3 Reconnaissance en ligne ou hors ligne

Deux approches de reconnaissance se distinguent selon le mode d'acquisition des données. Dans un système de reconnaissance hors ligne, les données sont considérées comme une représentation statique de l'écriture. Ce mode permet d'obtenir un nombre important des caractères instantanément ; cependant des prétraitements coûteux sont nécessaires pour retrouver l'ordre de la lecture de ces caractères dans le document originale. Dans le cas d'un système de reconnaissance en ligne, les données sont acquises (à l'aide d'un numériseur ou d'un assistant personnel par exemple) et reconnu en temps réel. Cela permet à l'usager de corriger et modifier son écriture d'une manière directe et instantanée.

2.2.4 Reconnaissance des caractères ou analyse de documents

Le système de reconnaissance peut changer selon la structure de données. Lorsque cette structure est limitée à des lignes et des mots, dans ce cas on parle de la reconnaissance des caractères. Il s'agit d'un simple repérage des lignes en mots puis découper ces lignes en caractères. Une structure plus complexe est celle des documents contenant des données textuelles et non textuelles en même temps. Dans ce cas, le document contient deux types de structures. La première décrit la logique du contenu d'un document au moyen d'entités logiques telles que les chapitres, les sections, les titres, les paragraphes, les notes, les citations, les formules, les tableaux, les cellules, les images et les graphiques. La seconde structure décrit l'entité physique et l'organisation hiérarchique des blocs typographiques constituant les pages d'un document, c'est-à-dire qu'elle décrit l'allure de la présentation dans les documents. Cette structure est nommée structure physique.

2.2.5 Systèmes avec apprentissage

Dans le cas général, les systèmes de reconnaissance des caractères sont formés en se basant sur des échantillons de données (écritures imprimées ou manuscrites). Ils contribuent énormément à la reconnaissance des données futures. L'étiquetage de ces échantillons est effectué manuellement ou automatiquement. Dans le premier cas et en connaissant l'étiquette de chaque échantillon, l'utilisateur aide le système à organiser ces échantillons en attribuant chacun à sa classe ; il s'agit de l'apprentissage supervisé. Dans le deuxième cas, des algorithmes sont introduits pour donner des étiquettes à ces échantillons, on parle de l'apprentissage non supervisé.

2.3 Architecture générale d'un système d'OCR

Notre travail est concentré sur l'élaboration de nouveaux systèmes de reconnaissance hors ligne des caractères. Un système de reconnaissance hors ligne des caractères est décomposé en trois étapes essentielles : prétraitements, extraction des caractéristiques et classification (voir figure 2.1 ci-dessous).

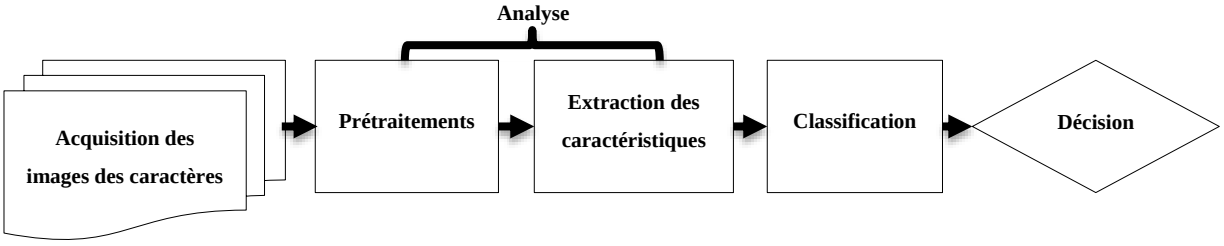


Figure 2-1. Systèmes de reconnaissance des caractères

Les images acquises sont analysées à fin d'extraire les primitives décrivant au mieux chaque image. En premier lieu, l'image est passée par un processus de prétraitements dont le but est d'améliorer la qualité de l'image en gardant les informations pertinentes et réduire la quantité de données dans les processus qui suivent. En second lieu, une description synthétique est produite à partir de l'image prétraitée. Cette description est exploitée par un processus de classification des caractères.

Dans la suite de cette section, nous allons présenter un état de l'art sur les trois étapes principales de la reconnaissance des caractères en respectant l'ordre présenté à la figure 2.1 ci-dessus.

2.3.1 Prétraitements

L'importance de l'étape de prétraitement d'un système de reconnaissance des caractères, réside dans sa capacité à remédier à certains problèmes qui peuvent se produire lors de l'étape d'acquisition. Ainsi, l'utilisation de ces techniques peut améliorer la qualité de l'image en la préparant pour l'étape suivante dans un système de reconnaissance des caractères.

Dans les systèmes de reconnaissance des caractères, la plupart des applications utilisent des images au niveau de gris ou noir et blanc, car le traitement des images en couleur est couteux. De telles images peuvent également contenir un arrière-plan rendant difficile l'extraction du texte du document à partir de l'image sans effectuer un quelconque prétraitement.

Le résultat souhaité du prétraitement est une image binaire contenant uniquement du texte. Pour y parvenir, plusieurs techniques sont mises à notre disposition. Certaines techniques servent à améliorer l'image en supprimant le bruit et / ou corriger le contraste. D'autres, utilisent le seuillage pour éliminer le bruit de l'arrière-plan. Ensuite, il y a des techniques de segmentation qui permettent de séparer les caractères les uns les autres. Et enfin, le traitement morphologique pour améliorer les caractères dans les cas où le seuillage ou d'autres techniques de prétraitement érodent des parties du caractère ou des pixels ajoutés.

Dans la suite nous présentons quelques techniques utilisées au cours de l'étape du prétraitement d'un système de reconnaissance des caractères.

2.3.1.1 Traitement ponctuel

Le traitement ponctuel modifie les valeurs des pixels de l'image originale pour créer les valeurs des pixels correspondants dans l'image améliorée, ce qui est exprimé dans l'équation 2.1.

$$O(x,y) = T[I(x,y)] \qquad (2.1)$$

Tel que, $I\ (x,y)$ est l'image d'origine (d'entrée), $O\ (x,y)$ est l'image améliorée et T décrit la transformation entre les deux images. Parmi les techniques de traitement ponctuel, les techniques de seuillage sont les plus utilisés.

2.3.1.1.1 Techniques de seuillage

Le seuillage d'image est le processus consistant à séparer les informations (objets) d'une image de son arrière-plan. Il est généralement appliqué aux images numérisées au niveau de gris ou en couleur. Le seuillage peut être classé en deux catégories principales: global et local. Les méthodes de seuillage global choisissent une valeur de seuil pour l'image entière du document, qui est souvent basée sur l'estimation du niveau de fond à partir de l'histogramme d'intensité de l'image. D'autre part, le seuillage local utilise des valeurs différentes pour chaque pixel en fonction de l'information de la zone locale.

2.3.1.1.1.1 Seuillage global

Dans la littérature, de nombreuses méthodes de seuillage ont été publiées, par exemple Saho et al. [3] ont comparé les performances de plus de 20 algorithmes de seuillage global en utilisant des mesures uniformes. La comparaison a montré que la méthode de séparabilité par classe Otsu [4] donnait les meilleures performances. D'autre part, dans une évaluation de la détection des changements par Rosin et Ioannidis [5], on a conclu que l'algorithme d'Otsu était très peu performant comparé à d'autres méthodes globales. L'étude d'OCR réalisée par Trier et Jain [6] a examiné quatre techniques globales qui montrent que la méthode d'Otsu a surpassé les autres méthodes étudiées. En outre, Fischer [7] a comparé 15 méthodes globales et a confirmé que la méthode Otsu est préférée dans le traitement d'images de document. La méthode Otsu est l'une des techniques largement utilisées pour convertir une image au niveau de gris en une image binaire puis calcule le seuil optimal séparant ces deux classes afin que leur propagation combinée (variance intra classe) soit minimale.

2.3.1.1.1.2 Seuillage local

Les techniques de seuillage locaux sont utilisées avec les images de document ayant des arrière-plans complexes. Dans la littérature, les techniques de seuillage locales sont développées principalement pour des applications spécifiques et la plupart du temps elles ne fonctionnent pas bien dans différentes applications. Les résultats peuvent être supérieurs au seuillage ou sous seuillage en fonction du contraste et de la luminosité. D'après la littérature, plusieurs études ont comparé différentes techniques de seuillage. Le travail de Treves et Jain [6] a évalué la performance de 11 méthodes de seuillage localement adaptées et bien établies. Ces techniques ont été comparées en utilisant un critère basé sur la capacité d'un module OCR

à reconnaître des chiffres manuscrits à partir d'images hydrographiques. Dans cette évaluation, la méthode de Niblack [6], semble être la meilleure. Cependant, si différents ensembles d'images sont utilisés avec différentes méthodes d'extraction de caractéristiques et de classificateurs, alors cette observation peut ne pas être exacte et une autre méthode pourrait surpasser la méthode de Niblack.

2.3.1.1.2 Traitement d'histogramme

Le traitement d'histogramme est utilisé dans l'amélioration, la compression et la segmentation de l'image. Un histogramme trace simplement la fréquence à laquelle chaque niveau de gris se produit de 0 (noir) à 255 (blanc). Les images numérisées ou capturées peuvent avoir une gamme limitée de couleurs ou ne pas avoir de contraste (détails). L'amélioration de l'image par traitement d'histogramme peut permettre d'améliorer les détails, mais peut également aider d'autres opérations à vision par ordinateur, telles que la segmentation. C'est une étape initiale du prétraitement. L'égalisation d'histogramme [8] et la spécification d'histogramme [9] (correspondance) sont deux méthodes largement utilisées pour modifier l'histogramme d'une image pour produire une image bien meilleure.

2.3.1.2 Réduction de bruit

Malgré le développement technologique des outils d'acquisition d'images, il existe encore des sources de bruit qui ne peuvent être éliminées. Il s'agit des variations stochastiques par opposition aux distorsions déterministes, telles que l'ombrage ou le manque de focalisation. La plupart du bruit peut être éliminé par les capteurs. Les systèmes d'analyse de documents bénéficient de la réduction du bruit en phase de prétraitement ce qui peut apporter une amélioration substantielle de la fiabilité et de la robustesse des étapes d'extraction des caractéristiques et de classification du système OCR. Une manifestation commune du bruit dans les images binaires prend la forme de pixels isolés. Le traitement de l'élimination de ce type de bruit est appelé remplissage [10]. Dans les images en gris, les filtres médians et les filtres passe-bas ont permis d'éliminer le bruit de pixel isolé. Le flou gaussien et les filtres moyens sont un meilleur choix pour donner une texture lisse à l'image. D'autre part, le bruit périodique qui se manifeste sous forme de rafales impulsionnelles qui sont souvent visibles dans le spectre de Fourier peut-être filtré en utilisant un filtrage d'encoche.

2.3.1.3 Détection et correction d’inclinaison

En raison de la possibilité de rotation de l'image d'entrée, le biais de document doit être corrigé. Les techniques de détection d’inclinaison visent à retrouver la structure des lignes horizontales dans un document texte. Hull et Taylor [11], ont enquêté sur vingt-cinq méthodes différentes pour la détection de biais de l’image. Les méthodes comprennent des approches basées sur l'analyse des transformations d'Hough, le profil de projection, la distribution de points des caractéristiques et l'analyse des caractéristiques sensibles à l'orientation. L'enquête a permis de conclure que la plupart des techniques indiquaient une amplitude allant jusqu'à 0,1 degré de précision, ce qui démontre la nécessité de poursuivre les travaux dans ce domaine pour montrer les forces et les faiblesses des algorithmes individuels. En outre, de nouvelles techniques émergent pour des applications spécifiques telles que la méthode d'Al-Shatnawi et d'Omar [12] qui est basée sur le centre de gravité pour traiter des images de documents arabes. Par conséquent, le choix d'utiliser une technique de détection / correction d'inclinaison, dépend de l'application et du type d'image utilisé.

2.3.1.4 Segmentation des caractères

La segmentation des caractères est considérée comme l'une des principales étapes du prétraitement, en particulier dans les scripts cursifs tels que l'Arabe et d'autres scripts où les caractères sont connectés ensemble. Par conséquent, il existe de nombreuses techniques développées pour la segmentation des caractères et la plupart d'entre elles sont spécifiques au script et ne peuvent pas fonctionner avec d'autres scripts. Même dans les documents manuscrits imprimés, la segmentation des caractères est nécessaire en raison de la cursivité de l’écriture manuscrite. Par exemple, les caractères latins imprimés sont faciles à segmenter en utilisant des profils d'histogrammes horizontaux et verticaux; cependant, les polices plus petites et celles contenant des serifs peuvent introduire un chevauchement qui nécessitera un traitement ultérieur pour résoudre le problème.

2.3.1.5 Opérations morphologiques

Les résultats de segmentation peuvent entraîner la suppression de certains pixels qui produisent des trous dans certaines parties des images; Cela pourrait être vu à partir des caractères ayant certains trous causés par la suppression de quelques pixels pendant le seuillage. Des trous plus grands peuvent provoquer des ruptures des caractères en deux ou

plusieurs parties. D'autre part, comme la segmentation peut joindre des objets séparés rendant plus difficile la séparation des caractères; ces objets solides ressemblent à des gouttes et sont difficiles à interpréter. La solution à ces problèmes est le filtrage morphologique. Les techniques [13] utiles incluent l'érosion et la dilatation, l'ouverture et la fermeture, le contour, l'amincissement et la squelettisation. Ces techniques ne fonctionnent que sur des images binaires.

2.3.1.5.1 Érosion et Dilatation

Les opérations de dilatation et érosion sont des opérations morphologiques qui augmentent ou diminuent les objets en taille et peuvent être très utiles pendant le prétraitement. L'érosion rend un objet plus petit en supprimant les pixels sur ses bords; cependant, la dilatation rend un objet plus grand en ajoutant des pixels autour de ses bords. Il existe deux techniques générales d'érosion et de dilatation: le seuillage et les techniques de masquage. La technique de seuillage regarde les voisins d'un pixel et change son état si le nombre de pixels voisins différents dépasse un seuil. Fondamentalement, si le nombre des pixels zéro au voisinage d'un pixel dépasse un paramètre de seuil alors le pixel est mis à zéro. La technique de masquage utilise un tableau $n \times n$ des uns et zéros sur une image d'entrée et érode ou dilate l'entrée, la direction de l'érosion ou de la dilatation peut être contrôlée [11].

2.3.1.5.2 Ouverture et Fermeture

Les opérations « ouverture » et « fermeture », désignent les opérateurs morphologiques qui sont dérivés des opérations fondamentales de l'érosion et de la dilatation. Ils sont normalement appliqués aux images binaires. L'effet de base d'une ouverture/fermeture est similaire à celui de l'érosion/dilatation dans la mesure où il tend à supprimer/ajouté certains des pixels du premier plan sur les bords. Cependant, il est moins destructeur que l'érosion/dilatation en général.

Les opérateurs d'ouverture et de fermeture fonctionnent bien, mais produisent parfois des résultats indésirables où la fermeture peut fusionner des objets qui ne doivent pas être fusionnés et l'ouverture peut élargir les trous et provoquer la rupture d'un objet [11, 13, 14].

2.3.1.5.3 Amincissement et squelettisation

La squelettisation est un processus qui permet de produire une version réduite d'une image binaire avec une épaisseur d'un seul pixel sans perdre les propriétés topologiques, de connectivité de l'image originale. Pendant le processus, la plupart des pixels d'avant-plan de l'image originale sont éliminés. Les pixels non éliminés sont ceux qui appartiennent au squelette de l'avant-plan de l'image.

Il existe deux techniques de base pour produire le squelette d'un objet: l'amincissement de base et les transformations de l'axe médian. C'est une opération morphologique qui est utilisée pour éliminer les pixels sélectionnés du premier plan des images binaires, un peu comme l'érosion ou l'ouverture. L'amincissement est un processus de réduction de données qui érode un objet jusqu'à ce qu'il soit d'un pixel de largeur, produisant un squelette de l'objet, facilite sa reconnaissance, tels que les caractères. D'autre part, la transformation de l'axe médian trouve les points d'un objet qui forment des lignes vers le centre [15]. Elle est semblable à la mesure de la distance euclidienne de n'importe quel pixel au bord de l'objet [11]. Le pixel qui minimise cette distance appartient à l'axe médian de l'objet.

2.3.2 Extraction des caractéristiques

L'extraction des caractéristiques peut être définie comme le processus d'élimination des informations redondantes à partir des données brutes ; ce qui permet de garder que les informations les plus représentatives. Ainsi, un ensemble de caractéristiques est extrait pour chaque classe permettant de la distinguer des autres classes, tout en restant invariant aux différences des caractéristiques dans la classe [16]. Trier et al. [17] a présenté une bonne enquête sur les méthodes d'extraction des caractéristiques pour la reconnaissance des caractères.

Généralement, il existe deux types de caractéristiques : caractéristiques statistiques [18] et caractéristiques structurelles [19]. Les caractéristiques statistiques sont bien fondées mathématiquement. Elles décrivent un motif en un ensemble de calculs statistiques ; cette description donne des informations locales sur le contenu de l'image. Les caractéristiques structurelles décrivent les propriétés topologiques et géométriques dans l'image. C'est une description très informative et discriminante, ce qui permet de prendre des décisions rapides dans la phase de reconnaissance avec une complexité de calcul modéré.

2.3.2.1 Description statistique

La représentation d'une image de document par répartition statistique des points prend en charge les variations de style dans certaines mesures. Bien que ce type de représentation ne permette pas la reconstruction de l'image originale, il est utilisé pour réduire la dimension du jeu des caractéristiques fournissant une vitesse élevée et une faible complexité. Un certain nombre de techniques sont utilisées pour l'extraction de caractéristiques; Certains d'entre eux sont: les moments, le zonage, les histogrammes de projection et les n-tuples.

2.3.2.1.1 Zonage

Le principe du zonage est de diviser le cadre contenant le caractère en plusieurs zones qui se chevauchent ou non. Les densités des points ou de certaines caractéristiques dans différentes régions sont analysées [20].

Les topologies de zonage peuvent être classées en deux grandes catégories: Statique et Adaptative. Les topologies statiques sont conçues sans utiliser d'informations sur la distribution des entités dans les classes des motifs. Dans ce cas, la conception de zonage est réalisée selon des preuves expérimentales ou sur la base de l'intuition et l'expérience du concepteur. Inversement, les topologies adaptatives peuvent être considérées comme les résultats des procédures d'optimisation pour la conception de zonage. Dans ce cas, une variété d'informations peut être utilisée pour concevoir la topologie la plus rentable pour la classification spécifique du problème.

2.3.2.1.2 Moments

En générale, les moments décrivent des quantités numériques à une certaine distance d'un point de référence ou d'un axe. Dans le cas des OCR, les moments de différents points présents dans un caractère sont utilisés comme caractéristiques. Ils sont considérés comme les méthodes les plus utilisées dans la reconnaissance des caractères. L’une des premiers concepts est celui des moments invariants de Hu [21] ; il s’agit de mesurer la distribution des pixels autour du centre de gravité du caractère. Ensuite, les moments de Zernike ont été proposés par Teh et al. [22], ils exigent une précision de calcul inférieure pour représenter les images avec la même précision que les moments réguliers. D’autres moments se basent sur la distance des différents pixels présents dans la matrice des caractères à partir du centroïde du caractère (moments centraux) [23].

2.3.2.1.3 Histogrammes de projection

Les histogrammes de projection nous donnent le nombre de pixels noirs dans les directions verticale et horizontale de la zone spécifiée du caractère. Ce concept a été introduit par M. H. Glauberman en 1956 [24] dans un système OCR. Les histogrammes de projection peuvent être verticaux, horizontaux, diagonaux à gauche ou diagonaux droits.

2.3.2.1.4 N-tuples

Cette méthode définie la position des pixels noirs ou blancs dans une image du caractère comme une caractéristique. Elle a été développée par Tarling et Rohwer en 1993 [25]. Cette méthode fournit un certain nombre de propriétés importantes de pixels.

2.3.2.2 Description structurelle

Les descriptions structurelles consistent à extraire les propriétés topologiques et géométriques du caractère avec une tolérance élevée aux distorsions et aux variations de style. Ce type de représentation peut également encoder certaines connaissances sur la structure de l'objet ou peut fournir quelques connaissances sur les composants constituant cet objet. Diverses représentations topologiques et géométriques peuvent être regroupées en quatre catégories: extraction de structures topologiques, mesure des propriétés géométriques, codage et représentation par graphes.

2.3.2.2.1 Extraction de structures topologiques

Dans cette catégorie, les formes géométriques simples et leurs attributs (longueur, position, orientation…) sont utilisés comme description de la forme du caractère [26, 27, 28, 29]. Cette description est généralement extraite à partir du squelette ou du contour du caractère [30]. Parmi ces formes géométriques, les coins [31] qui sont des éléments locaux importants dans les images. En général, ce sont les points qui ont une courbure élevée et qui se trouvent dans la jonction des différentes régions de luminosité des images. Dans une variété de caractéristiques d'images, les coins ne sont pas affectés par l'illumination et ont la propriété de l'invariance rotationnelle. Ils ne représentent que 0,05% environ dans l'ensemble des pixels.

Sans perdre des informations de l'image, l'extraction de coins peut minimiser les données de traitement. Par conséquent, la détection de coin a une valeur pratique et il joue un rôle important dans la représentation de l'image [32] et dans d'autres domaines.

Un grand nombre de détecteurs de coin ont été proposés par les chercheurs. Ces méthodes peuvent être divisées en deux classes principales: une est basée sur le contour et l'autre est basé sur l'intensité. Les méthodes basées sur le contour récupèrent d'abord les contours de l'image, puis recherchent les maximas de courbure ou les points d'inflexion le long de ces contours. Par exemple, Peng et al. [33] ont introduit une méthode de détection d'angle basée sur les contours utilisant la transformée en ondelettes pour sa capacité à détecter des variations brusques. Les méthodes basées sur l'intensité, estiment une mesure qui est destinée à indiquer la présence d'un coin directement à partir des valeurs de niveaux de gris dans une image [34, 35, 36, 37]. Cette catégorie se caractérise par sa rapidité et son indépendance par rapport à d'autres caractéristiques locales. Deux algorithmes représentatifs de cette catégorie : l'algorithme de détection de coins Susan [38] et l'algorithme de détection de coins Harris [39], qui sont les algorithmes de détection de coins les plus utilisés en pratique. Une description détaillée de ces deux algorithmes serai abordé dans les chapitres qui suivent.

2.3.2.2.2 Mesure des propriétés géométriques

Dans cette catégorie, les caractères sont représentés par la mesure des grandeurs géométriques telles que : le rapport entre la largeur et la hauteur du plus petit rectangle contenant le caractère, l'horizontale relative, les distances verticales entre le premier et le dernier point, la distance entre deux points que ce soit euclidienne ou riemannienne, les longueurs comparatives entre deux traits, la largeur d'un trait, les masses supérieures et inférieures de mots, la courbure de la longueur du mot ou la variation de la courbure [40].

2.3.2.2.3 Codage

Le codage (chaine de codes) est basé sur l'idée qu'une représentation d'un contour ou squelette peut être obtenue en stockant la position relative entre les pixels consécutifs dans l'image. Les techniques de codage sont l'une des plus anciennes techniques de vision par ordinateur, initialement introduites par Freeman [41]. Essentiellement, l'ensemble de pixels dans la bordure ou le squelette d'une forme est traduit en un ensemble de connexions entre eux. Étant donné un ensemble de points connectés, puis à partir d'un pixel, nous devons être capables de déterminer la direction dans laquelle le pixel suivant doit être trouvé. À savoir, le pixel suivant est l'un des points adjacents dans l'une des directions principales du compas. Ainsi, le code de chaîne est formé en concaténant le nombre qui désigne la direction du pixel

suivant. C'est-à-dire, étant donné un pixel, la direction successive d'un pixel au pixel suivant devient un élément dans le code final. Cela est répété pour chaque point jusqu'à ce que le point de départ soit atteint lorsque la forme (fermée) est complètement analysée.

2.3.2.2.4 Représentation par les graphes

Le but est de représenter la forme par un graphe. Cette représentation est basée sur l'idée que toute forme est appréhendée par la perception visuelle humaine et non pas comme un ensemble, mais en certaines parties, leurs positions spatiales et leurs relations, jouent un rôle très important dans l'apprentissage et la reconnaissance.

2.3.3 Classification

En reconnaissance des formes, la classification représente le problème d'attribution d'une forme inconnue à une classe prédéfinie. Elle est considérée comme une instance de l'apprentissage supervisé où un ensemble d'observations correctement identifiés est disponible. Généralement, les algorithmes de classification définis la fonction mathématique qui permet de mapper des données d'entrée à une classe/catégorie.

Les algorithmes de classification peuvent être catégorisés selon l'approche adoptée pendant la réalisation d'un système de reconnaissance des formes. Il existe deux approches fondamentales pour mettre en œuvre un système de reconnaissance des formes: statistique et structurelle. Chaque approche utilise différentes techniques pour mettre en œuvre les tâches de description et de classification. Les approches hybrides, parfois appelées une approche unifiée de la reconnaissance des formes, combinent à la fois des techniques statistiques et structurelles au sein d'un système de reconnaissance des formes.

2.3.3.1 Approche statistique

Il est bien connu que l'approche statistique représente des objets du monde réel au moyen d'un ensemble de mesures : une fois que ses caractéristiques (disons n) ont été extraites, un objet devient un point dimensionnel dans l'espace du vecteur correspondant. La raison réside dans le fait que les propriétés mathématiques des espaces vectoriels sont utilisées pour faire face au problème de l'apprentissage et de la classification des modèles par des algorithmes intelligents et bien compréhensibles. En effet, lorsque les caractéristiques utilisées ont été adéquatement sélectionnées (par exemple par analyse discriminante), une propriété

importante est censée de tenir que deux points, proches l'un de l'autre dans l'espace vectoriel, correspondent à des objets similaires dans le monde réel. Il convient de souligner l'impact pratique de cette propriété de base : l'espace vectoriel comprend une mesure de distance telle que la distance euclidienne comme outil clé pour mesurer la distance entre les points et il peut être utilisé comme une similitude des objets correspondant à ces points. Ainsi, le problème d'évaluer la (dis)-similarité entre deux objets sont simplement ramenés à l'évaluation de la distance entre les vecteurs de caractéristiques correspondantes (voir figure 2.2).

Les techniques statistiques utilisées pour la classification incluent celles basées sur la similarité (comme l'appariement de modèle), la probabilité (réseaux bayésiens, arbres de décisions), les limites (réseaux de neurone, SVM) et les regroupements (k-moyennes, k-means, k-plus proches voisins (K-NN)). Ces techniques sont largement présentées dans plusieurs thèses.

2.3.3.2 Approche structurelle

Bien que les approche statistiques sont bien fondées mathématiquement et bien généralisées, ils traitent les objets de façon abstraite sans prendre en compte la nature des mesures effectuées. En contrepartie, les approches structurelles mettent en importance la structure physique des formes en tenant compte leurs propriétés topologique et géométrique. Généralement l'extraction du squelette ou la détection des contours de la forme est prérequise à fin de la décomposer en plusieurs primitives qui sont généralement des formes géométriques simples telles que les point, les segments droite, les cercles, les arrêtes, etc. il est important de noter que les positions de ces primitives ainsi que leur relation jouent un rôle très important dans l'apprentissage et la reconnaissance. Ces relations sont fréquemment représentées par des graphes, des chaînes ou des grammaires respectant des règles spécifiques de syntaxe [42]. Dans ces approches, on distingue deux catégories de méthodes : méthodes d'appariement des graphes, méthodes d'appariement des chaînes.

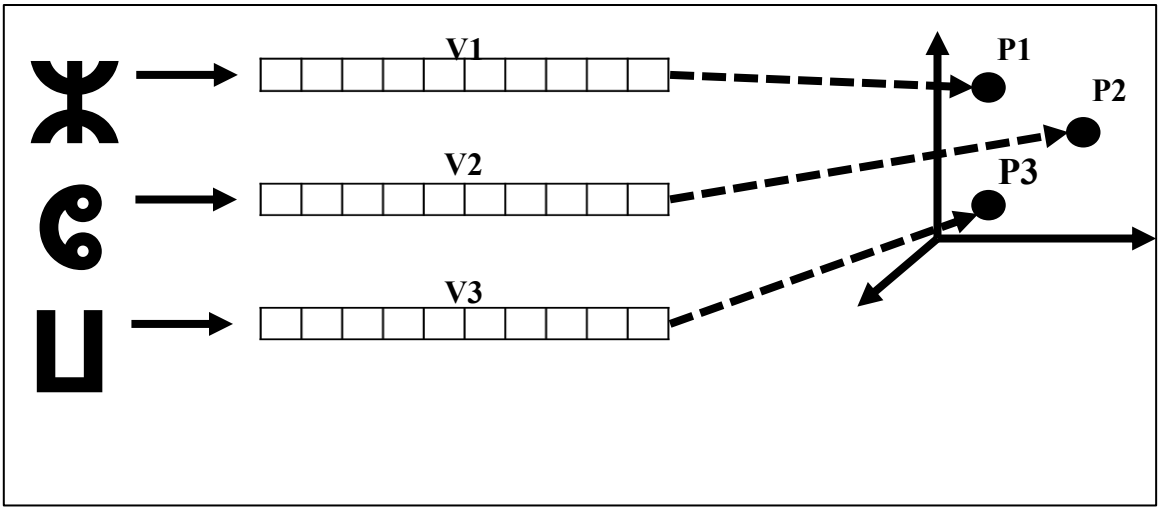


Figure 2-2. Représentation des objets en reconnaissance des formes statistique ; les vecteurs caractéristique sont extraits puis les objets sont représentés comme des points dans l'espace de ces vecteurs

2.3.3.2.1 Appariement des graphes

L'utilisation des graphes dans la reconnaissance des formes remonte au début des années 1970 ; ils ont été utilisés comme un outil puissant pour représenter et classer les motifs en se basant sur leurs caractéristiques géométriques, en particulier dans les méthodes structurelles. Il s'agit de voir les objets comme un ensemble de parties convenablement reliées les unes aux autres. Dans cette hypothèse, les nœuds des graphes, enrichis les attributs correctement définis et peuvent être utilisés comme descripteurs des parties composantes des objets, alors que les arrête des graphes représentent les relations entre les parties. Plusieurs études sur les techniques basées sur les graphes ont été publiées jusqu'à présent sur des différents sujets, tels que : la représentation des graphes, la mise en correspondance des graphes, la distance d'édition des graphes, l'encapsulation des graphes et les graphes noyaux [43-49]. Plus récemment Foggia et al. [50] introduit une catégorisation détaillée des méthodes basées sur des graphes durant les 40 dernières années.

Une étude détaillée sur l'utilisation des graphes en reconnaissance des formes est abordé dans le chapitre qui suit.

2.3.3.2.2 Appariement des chaines

Dans cette catégorie, les motifs sont représentés sous forme d'une chaine de primitives. Les algorithmes d'appariement des chaines consistent à calculer la distance d'édition [51-52] entre

un modèle de référence et le motif à reconnaitre. Il s'agit de calculer le nombre des opérations pour transformer les chaines du caractère d'un modèle de référence à un modèle du caractère à reconnaitre.

2.3.3.3 Approche hybride

L'idée de combiner plusieurs classificateurs a été proposée comme une nouvelle direction pour obtenir des systèmes de reconnaissance fiable. Le but est d'utiliser plusieurs classificateurs pour compenser les faiblesses des uns aux autres. Cela a été approuvé en application sur plusieurs types de motifs, surtout les caractères manuscrits [53,54]. Une analyse détaillée sur les méthodes hybride et leurs applications sur l'écriture a été proposée par Zouari, [55].

Selon la façon de combiner les classificateurs on distingue trois type de combinaison (voir figure 2.5 ci-dessous) : une combinaison parallèle qui consiste à utiliser chaque classificateur indépendamment puis combiner les sorties pour donner le résultat de la reconnaissance. Une combinaison séquentielle dont chaque classificateur est situé à un niveau à fin de réduire le nombre des classes possible jusqu'à atteindre le résultat final. Et enfin, une combinaison hybride consiste à utiliser la combinaison parallèle et séquentielle dans une seule architecture pour résoudre les problèmes de classification complexe.

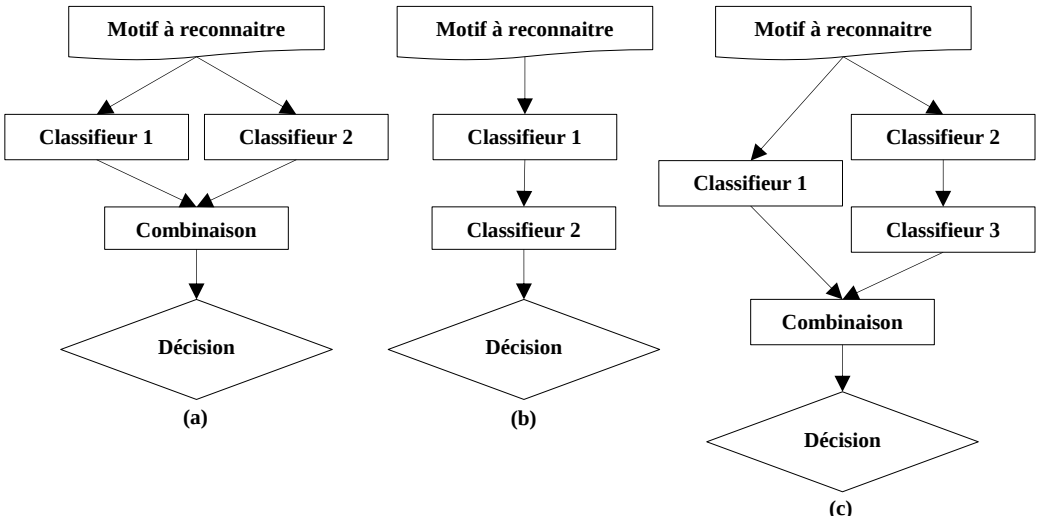


Figure 2-3. Les trois architectures de combinaison des classificateurs : (a) combinaison parallèle, (b) combinaison séquentielle, (c) combinaison hybride

2.4 État de l'art de la reconnaissance des caractères Tifinaghs

Étant donné que l'intégration du système d'écriture de la langue Amazighe dans les systèmes internationaux est assez récente. Les projets de recherches sur la reconnaissance des caractères de cette langue sont rares comparés aux autres langues telles que l'Arabe et le Latin. Dans la suite de cette section, nous citons les travaux de recherches qui concernent la reconnaissance optique des caractères Tifinaghs imprimés et manuscrits.

Parmi les premiers travaux sur la reconnaissance optique des caractères Tifinaghs imprimés nous citons :

- Oulamara et al. [56] ont utilisé la transformée de Hough pour extraire les segments droits avec leurs attribues (langueur et degré d'inclinaison). En analysant les caractères dans l'espace paramétrique, une matrice de lecture est construite contenant les vecteurs caractéristiques des images de référence dans la phase d'apprentissage. En utilisant une base de données locale, les auteurs ont obtenu des résultats intéressants. Cependant, selon Djematene et al. [57], cette technique n'est pas appropriée car la segmentation par la transformée de Hough ne produit pas une segmentation correcte.
- Ait Ouguengay et al. [58] ont proposé un système de reconnaissance basé sur les réseaux de neurones artificiels (RNA) multicouche à une seule couche caché pour classifier les vecteurs caractéristiques composés des propriétés géométrique (les projections horizontales et verticales, les centres de gravité en x et en y, le périmètre, l'aire, la compacité et les moments centraux d'ordre 2). Des résultats intéressants sont obtenus en testant le système sur une base de données locale.
- Amrouch et al. [59] ont proposé une approche basée sur l'extraction d'informations directionnelles à partir de la transformation de Hough de chaque caractère sous la forme d'un vecteur d'observations. Cette information alimente un modèle de Markov caché (HMM). Les résultats obtenus sont prometteurs. Cependant, la discrimination de ces modèles n'est pas très bonne parce que dans la phase d'apprentissage chaque caractère est représenté par une seule image de référence. À fin de remédier à cette issue, Amrouch et al. [60] ont remplacé la transformé de Hough par une nouvelle technique pour exprimer un ensemble de caractéristiques structurelles à partir du contour du caractère basé sur des points qui ont une déviation maximale. Dans la phase d'apprentissage et classification, ils ont combiné la programmation dynamique avec les HMM continus. Cette approche, a

l'avantage d'être indépendante du nombre de classes de reconnaissance (en termes de mémoire et de vitesse) puisque le modèle est construit pour toutes les classes. Les résultats, qui sont tout à fait encourageants, ont montré que les HMM continus sont plus robustes. Cependant, les inconvénients de cette approche sont la détection des points qui ont la déviation maximale pour la phase d'extraction des caractéristiques qui semblent restrictives pour certaines polices de l'écriture Amazighe.

- Dans une tentative de résolution des problèmes d'orientation et de changement de taille, El Ayachi et al. [61] ont comparé deux descripteurs robustes. Il s'agit des moments invariants et la transformée de Walsh. Les auteurs ont présenté deux systèmes contenant les mêmes processus de prétraitement et de classification. À l'aide de la programmation dynamique dans la phase de classification, les auteurs ont conclu que les moments invariants sont supérieurs à la transformée de Walsh en terme du temps d'exécution et de la discrimination. À fin d'améliorer le taux de reconnaissance, El Ayachi et al. [62] ont remplacé la méthode de programmation dynamique par un réseau de neurones à une seule couche caché. Le système a donné un meilleur taux de reconnaissance comparé à la programmation dynamique et aux réseaux de neurones à 2 ou 3 couches cachés. Récemment, dans [63] les auteurs ont utilisés leurs systèmes à fin d'afficher le code braille adapté aux caractères Tifinaghs. Le système braille est un système adopté pour aider les aveugles et les malvoyants à s'intégrer dans les différents secteurs de la vie.
- Es Saady et al. [64] ont proposé une approche syntaxique pour la reconnaissance des caractères Tifinaghs imprimés. Cela est fait en représentant les images des caractères à l'aide du codage de Freeman. La classification est effectuée en utilisant les automates finies. Une automate est construite à partir des automates spécifiques de chaque image du caractère Tifinagh imprimé. Selon les auteurs, les tests effectués montre la robustesse du système. Cependant, le problème de cette approche est qu'elle ne traite pas les caractères circulaires. Pour y remédier, les auteurs ont utilisé la symétrie horizontale et verticale de la graphie de l'alphabet tifinagh. En effet, les auteurs ont présenté dans [65] un système basé sur la position des lignes centrales de chaque caractère. Les caractéristiques sont extraites à base de la densité des pixels contenus dans une fenêtre glissante dans l'image du caractère. Selon les auteurs, cette approche a prouvé sa puissance de discrimination en la testant sur une base de données locale.

- À fin de complémenter aux limites des systèmes précèdent face aux problèmes de rotation et de changement de taille, Bencharef et al. [66] ont proposé une approche basée sur une description géométrique utilisant les descripteurs géodésiques. Ces descripteurs ont servie comme entré au processus de classification hybride qui combine les réseaux de neurones et les arbres de décision.
- Le succès de l'approche de Bencharef dans [66], a motivé Oujaoura et al. pour aller dans le même sens. Dans une première approche [67], les auteurs ont effectué une comparaison entre la transformée de Walsh, GIST et texture en utilisant les réseaux bayésiennes comme classificateurs. Les tests de ces descripteurs sur une base de données locale ont montré la supériorité de la méthode GIST en termes de taux de reconnaissance et temps de calcul. Dans une deuxième approche [68], les auteurs ont proposé un système qui combine les moments de Zernike, les moments de Legendre, les moments de Hu, la transformée de Walsh et GIST dans la phase d'extraction des caractéristiques; Ainsi que les réseaux de neurone, SVM et NN dans la phase de classification. Les résultats obtenus sont excellents en termes de taux de reconnaissance. Cependant, le système est assez lent en termes de temps de calculs.

En ce qui concerne la reconnaissance optique des caractères Tifinaghs manuscrits, à notre connaissance, juste quatre travaux de recherche sont réalisés dans ce contexte :

- Amrouch et al. [69] ont décrit une approche globale pour la reconnaissance du caractère manuscrit de Tifinagh dans laquelle ils ont utilisé la transformation d'Hough comme méthode d'extraction des caractéristiques et les modèles de Markov cachés (HMM) pour la classification. Cette approche n'a pas permis d'obtenir une discrimination performante entre les caractères manuscrits.
- El Kessab et al. [70] ont combiné les perceptrons multicouches et HMM pour réaliser un système plus discriminatif. Les caractéristiques sont extraites à l'aide des opérations morphologiques. Cette approche est performante en termes de taux de reconnaissance mais plus lent en termes de temps d'exécution.
- Gounane et al. [71] ont proposé une approche hybride en combinant les K-NN et b-Gram dans la phase de classification et en utilisant la distance des pixels par rapport au centre de gravité et leur densité dans la phase d'extraction des caractéristiques. Le système a produit

des résultats satisfaisants, mais le taux de reconnaissance peut être amélioré en utilisant un corpus plus grand.

- Es-saady et al. [72] a présenté un système où la position horizontale et verticale de base sont les caractéristiques de l'image du caractère. À l'aide des automates finis, ces caractéristiques sont classifiées. Le système a obtenu de meilleurs résultats lorsque les lignes de base ont été remplacées par la ligne centrale.

Les systèmes de reconnaissance cités ci-dessous, sont basés sur une approche statistique ou syntaxique de reconnaissance des formes. Les méthodes de classification adoptées sont performantes, cependant les méthodes d'extraction des caractéristiques adoptées ne satisfaitent pas les contraintes de la variation intra classe et la ressemblance inter classe.

2.5 Conclusion

Dans ce chapitre, nous avons présenté, dans un premier temps, les différents définitions et notions relatives à la chaine de processus de reconnaissance optique des caractères. En effet, nous avons décrit les différents éléments et aspects de l'OCR en se basant sur un diagramme comportant les étapes de sa réalisation. Nous avons présenté chaque processus du diagramme (prétraitement, Extraction des caractéristiques et classification) en catégorisant les différentes méthodes développés et utilisés dans la reconnaissance des caractères.

Dans un deuxième temps, nous avons cité les différents travaux réalisés dans le domaine de la reconnaissance hors ligne des caractères Tifinaghs. La totalité des systèmes réalisés dans ces travaux s'intègrent dans la catégorie des approches statistiques et géométriques de reconnaissance des formes. La majorité de ces travaux se concentrent sur l'amélioration de taux de reconnaissance en négligeant le facteur de temps de calcul.

Choisir une approche pour la reconnaissance d'une certaine écriture dépend énormément de ces caractéristiques morphologiques, de la taille de la base de références et du temps de calculs requis. Dans ce travail nous avons opté pour l'utilisation des graphes pour la reconnaissance des caractères Tifinaghs.

Dans le chapitre qui suit, nous allons introduire les différents algorithmes d'appariement des graphes utilisés dans la reconnaissance des formes, en décrivant la façon avec laquelle ils ont été utilisés et leurs développements.

3 Reconnaissance des formes par les graphes

3.1 Introduction

3.1.1 Éléments de base de la théorie des graphes

Dans cette section, on va définir quelques éléments de base sur les graphes dans la reconnaissance des formes.

Intuitivement, un graphe représente un ensemble d'éléments et un ensemble de relations entre ces éléments. Les éléments sont appelés nœuds ou sommets, et les relations sont appelées arrêtes. Formellement, un graphe G est défini par les ensembles $G = (N, A)$ dans lesquels $A \subseteq N \times N$. On peut désigner le $i-$ème sommet comme $n_i \in N$ et le $i-$ème arrête comme $a_i \in A$. Chaque arrête est un sous-ensemble de deux sommets, on peut aussi écrire $a_{ij} = \{n_i, n_j\}$.

Chaque graphe peut être considéré comme pondéré. Soit un graphe $G = (N, A)$, la pondération des sommets est une fonction $p : A \to \mathbb{R}$ et la pondération des arrêtes est une fonction $p : N \to \mathbb{R}$. Pour simplifier la notation, nous utiliserons p pour désigner à la fois la pondération des sommets et des arrêtes. Le poids d'un sommet est noté $p\,(n_i)$ ou p_i, et le poids d'un arrête incident à deux sommets est noté $p\,(n_i, n_j)$ ou p_{ij}. Si $p_i = 1, \forall n_i \in N \; et \; p_{ij} = 1, \forall a_{ij} \in A$, alors nous pouvons considérer le graphe comme non pondéré. Si on ne spécifie pas autrement, tous les poids des nœuds et des arrêtes sont considérés comme égaux à l'unité. Nous traitons $p_{ij} = 0$ comme équivalent à $p_{ij} \notin A$. Intuitivement, un poids nul d'arrête signifie que l'arrête n'est pas membre de l'ensemble des arrêtes.

Chaque arrête est considérée comme étant orienté. Une orientation d'un bord signifie que chaque arrête $a_{ij} \in A$ contient un ordre des sommets, v_i et v_j. Un bord a_{ij} est dirigé si $a_{ij} \neq a_{ji}$. Un graphe pour lequel aucun des bords ne sont pas dirigés est appelé un graphe non orienté, et un graphe dans lequel au moins une arrête est dirigé est appelé un graphe orienté ou un digraphe. Un bord dirigé est représenté par la notation $a_{i \to j}$. Un graphe orienté est plus général qu'un graphe non orienté car il n'exige pas que $a_{ij} = a_{ji}$ pour chaque arrête. Par conséquent, tous les algorithmes pour les graphes dirigés peuvent également être appliqués aux graphes non dirigés, mais l'inverse peut ou non être vrai.

Dessiner un graphe se fait typiquement en décrivant chaque nœud avec un cercle et chaque arrête avec une ligne reliant des cercles représentant les deux nœuds. L'orientation ou la direction du bord est généralement représentée par une flèche.

On considère deux fonctions $s, c: A \rightarrow N$. La fonction s est appelée fonction source, et la fonction c est appelée fonction cible. Soit $a_{ij} = (n_i, n_j) \in A$, on dit que $s(a_{ij}) = n_i$ est l'origine ou la source de a_{ij}, et $t(a_{ij}) = n_j$ est le point final ou cible de a_{ij}. Si l'on considère tout bord $ek \in A$, les sommets $s(ek)$ et $c(ek)$ sont appelés frontières de ek, et l'expression $t(ek) - s(ek)$ est appelée frontière de ek.

Compte tenu de ces préliminaires, nous pouvons maintenant énumérer une série de définitions de base:

1. **Adjacent** : Deux nœuds n_i et n_j sont appelés adjacents si $\exists a_{ij} \in A \; ou \; \exists a_{ji} \in A$, ce qui est noté $n_i \sim n_j$. Deux arrêtes a_{ij} et a_{sk} sont adjacentes si elles partagent un sommet commun, c'est-à-dire si $i = s, i = k, j = s$ ou $j = k$.
2. **Incident** : Un bord a_{ij} est incident aux nœuds a_i et a_j (et chaque noeud est incident au bord).
3. **Isolé** : Un nœud vi est appelé isolé si $p_{ij} = p_{ji} = 0, \forall n_j \in N$. Intuitivement, un nœud est isolé s'il n'est pas connecté au graphe par un bord (de poids non nul).
4. **Degré** : le degré d'un graphe est le nombre de nœuds le constituant. Le degré d'un nœud est les nombre d'arrêtes incident.
5. **Complément** : un graphe $\bar{G}(N, \bar{A})$ est dit complément d'un graphe $G(N, A)$ si l'ensemble $\bar{A} = N \times N - A$. Par conséquent, le graphe du complément possède les mêmes nœuds que l'original, mais chaque paire de nœuds dans le complément est connectée si le pair de nœuds n'a pas été connectée dans le graphe original.
6. **Régulier** : un graphe non orienté est dit régulier si tous ses nœuds ont le même degré : $\deg(n_i) = k, \; \forall n_i \in N$.
7. **Complet** : un graphe non orienté est appelé complet ou entièrement connecté si chaque nœud est connecté à chaque autre nœud par une arrête, c'est-à-dire lorsque $A = N \times N$. Un graphe complet de n sommets est noté K_n.
8. **Graphe partiel** : un graphe $G' = (N, A')$ est appelé un graphe partiel de $G = (N, A)$ si $A' \subseteq A$.

9. **Sous Graphe** : un graphe $G' = (N', A')$ est appelé sous graphe de $G = (N, A)$ si $N' \subseteq N$ et $A' = \{a_{ij} \in A | n_i \in N' \; and \; n_j \in N'\}$.
10. **Clique** : Une clique est définie comme un sous-ensemble entièrement connecté de l'ensemble de sommets.
11. **Bipartite** : Un graphe est appelé bipartite si N peut être divisé en deux sous-ensembles $N_1 \subset N \; et \; N_2 \subset N$, où $N_1 \cap N_2 = \emptyset \; et \; N_1 \cup N_2 = N$, tels que l'ensemble $A \subseteq N_1 \times N_2$. Si $|N_1| = n \; et \; |N_2| = m \; et \; A = N_1 \times N_2$, alors G est appelé graphe bipartite complet noté $K_{m,n}$.
12. **Isomorphisme de graphe** : Soit $G_1 = (N_1, A_1)$ et $G_2 = (N_2, A_2)$ deux graphes non orientés. Une bijection $f: N_1 \rightarrow N_2$ de G_1 à G_2 est appelée isomorphisme de graphe si $(n_i, n_j) \in A_1$ implique que $(f(n_1), f(n_2)) \in A_2$.

3.1.2 Représentation informatique des graphes

Plusieurs représentations informatiques des graphes peuvent être considérées. Les structures de données utilisées pour représenter les graphes peuvent avoir une influence significative sur la taille des problèmes qui peuvent être exécutés sur un ordinateur et la vitesse avec laquelle ils peuvent être résolus. Il est donc important de connaître les différentes représentations des graphes.

3.1.2.1 Représentation matricielle

Une représentation matricielle peut fournir un stockage efficace (puisque la plupart des graphes utilisés dans le traitement d'image sont creux), mais chaque représentation matricielle peut également être considérée comme un opérateur qui agit sur des fonctions associées aux nœuds ou aux arrêtes d'un graphe.

La première représentation matricielle d'un graphe est donnée par la matrice d'incidence. La matrice d'incidence pour $G = (N, A)$ est une matrice M de taille $|E| \times |V|$ où :

$$M_{ij} = \begin{cases} +1 \; si \; s(n_i) = n_j \\ -1 \; si \; s(n_j) = n_i \\ 0 \; si \; non \end{cases} \quad (3.1)$$

La matrice d'incidence peut conserver les informations d'orientation de chaque bord, mais pas le poids du bord. L'adjoint (transposé) de la matrice d'incidence représente également l'opérateur de frontière pour un graphe.

La représentation de la matrice d'adjacence du graphe $G = (N, A)$ est une matrice W de taille $|N| \times |N|$ où :

$$W_{ij} = \begin{cases} p_{ij}\ si\ a_{ij} \in A \\ 0\ si\ non \end{cases} \quad (3.2)$$

Cette représentation est considérée comme la plus compact et simple à calculer. Cependant, il y a une redondance d'information dans le cas des graphes non orientés.

3.1.2.2 Listes d'adjacence

L'avantage des listes d'adjacence sur les représentations matricielles est le moins d'utilisation de mémoire. En effet, une matrice d'incidence complète nécessite une mémoire $O(|N| \times |A|)$, et une matrice d'adjacence complète nécessite une mémoire $O(|N|^2)$. Cependant, les graphes peuvent tirer profit des représentations matricielles creuses pour un stockage beaucoup plus efficace.

Une représentation de liste d'adjacence d'un graphe est un tableau L de taille $|N|$ (Un pour chaque sommet de N). Pour chaque sommet n_i, il y a un pointeur L_i à une liste liée contenant tous les sommets n_j adjacents à n_i. Pour les graphes pondérés, la liste liée contient à la fois le sommet cible et le poids du bord. Avec cette représentation, l'itération à travers l'ensemble des arrêtes est $O(|N| + |A|)$ alors qu'elle est $O(|N|^2)$ pour les matrices d'adjacence. Cependant, vérifier si un bord est dans l'ensemble A, est une opération $O(|N|)$ avec des listes d'adjacence alors qu'elle est une opération O (1) avec des matrices d'adjacence.

3.1.3 La description graphique en reconnaissance des formes

L'utilisation d'une représentation à base des graphes induit la nécessité de formuler les principales opérations requises d'un système de reconnaissance en matière des graphes : l'apprentissage (qui est le processus pour obtenir un modèle d'une classe à partir d'un ensemble d'échantillons connus) et classification (conçue comme la comparaison entre un objet et un ensemble de prototypes). Ce sont quelques questions clés qui doivent être abordées à l'aide de techniques à base des graphes.

Les graphes représentent une structure de données beaucoup plus adéquate pour représenter des descriptions structurelles de motifs : les objets réels sont décomposés en parties, chacune décrite sous forme d'un ensemble de donnés de paramètres, et les relations entre ces parties. Les nœuds et les arrêtes des graphes sont ainsi associés aux parties et à leurs relations. La reconnaissance des formes structurelles parie sur cette structure de données, ce qui a donné beaucoup de succès à cette nouvelle approche auprès de comité de recherche Bunke et al. [73]

En plus de leur élégance représentationnelle, les graphes sont très adaptés pour exploiter la sémantique contextuelle au cours de la phase de classification : la pertinence d'une caractéristique donnée est souvent liée à sa position à l'intérieur de l'objet, c'est-à-dire de l'endroit où elle se trouve. Supposons, par exemple, d'avoir le caractère représenté sur la figure 3.1-a : tout le monde l'interprète comme un « un ». Si l'on ajoute à ce caractère un segment supplémentaire située à sa base, le caractère continue d'être interprété comme un « un » (voir figure 3.1-b) : le segment peut être très grand, comme sur la figure 3.1(c), mais, en raison de sa position, l'interprétation du caractère ne change pas : au contraire, si l'on ajoute même une très petite course approximativement au centre de la course verticale, comme dans la figure 3.1-d, l'interprétation change : le caractère devient un « sept ».

À partir de cette considération, il est immédiat de réaliser l'importance de l'utilisation de l'information contextuelle pendant la phase de reconnaissance. Les graphes, différemment des représentations à base de vecteurs, peuvent potentiellement traiter cet aspect, car la comparaison peut être effectuée d'une manière ou d'une autre par couplages successifs de paires de nœuds.

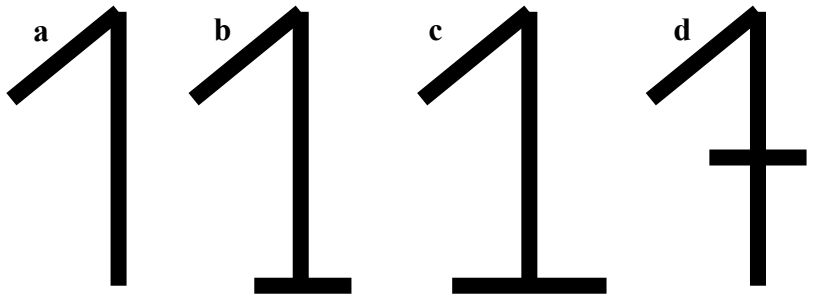


Figure 3-1. La pertinence du contexte. (a) Exemple du caractère « un ». (b) L'ajout d'un segment a la base du caractère ne change pas le contexte même s'il est grand comme en (c) Ajout d'un segment au milieu change le contexte en « sept »

3.2 Approches de reconnaissance des formes par les graphes

Dans la suite de ce chapitre, nous allons présenter un état de l'art sur la reconnaissance des formes par les graphes. Selon la façon dont ils ont été utilisés, on peut catégoriser les algorithmes d'appariement ou classification des graphes en trois approches : approche pure, approche impure et approche extrême.

- Approche pure: caractérisée par le fait que les problèmes de classification et d'apprentissage sont directement affrontés dans l'espace graphique. Les objets sont classés en comparant les graphes correspondants, en utilisant des algorithmes de correspondance adaptés. De même, l'apprentissage est approché par construction, soit manuellement ou automatiquement comme dans [74, 75]. Les prototypes de graphes sont considérés comme une généralisation d'un ensemble de graphes associés à des objets appartenant à la même classe.
- Approche impure: peuplée de méthodes qui transposent les opérations de base définies dans les espaces vectoriels sur des graphes, elle permet la réutilisation des méthodes d'apprentissage et de classification efficaces disponibles dans la vaste littérature sur les méthodes statistiques de reconnaissance des formes.
- Approche extrême : s'inspire de la transformation des graphes en vecteurs, de façon à appliquer les procédures d'apprentissage et de classement actuelles et futures définies dans les espaces vectoriels, comme les machines à noyaux modernes.

3.2.1 Approche pure

Vers les années 80, les intérêts des chercheurs en reconnaissance des formes par les graphes, étaient principalement concentrés sur la correspondance des graphes, exactes ou inexactes, ce dernier étant adapté pour traiter des graphes déformés (différents au niveau de la structure et/ou des nœuds et des arrêtes). Les méthodes exactes visent principalement à réduire le temps de calcul du processus d'appariement découlant de la complexité exponentielle de ce problème. Inversement, des méthodes inexactes tentent de trouver une sorte de coût d'appariement pouvant être une mesure de la similarité entre deux graphes différents ; les méthodes inexactes ont été inspirées par des approches différentes : des algorithmes optimaux, capables de garantir si une solution optimale existe, et des algorithmes sous

optimaux qui à leur tour assureront d'atteindre seulement un minimal local du coût d'appariement.

3.2.1.1 Appariement exact des graphes

Les premières tentatives pour réduire la complexité des calculs de l'appariement exacte des graphes ont été conçues pour des types particuliers de graphes. Parmi eux, nous trouvons des algorithmes pour certaines topologies courantes des graphes, comme les arbres (cas spéciaux de graphes), proposés par Aho et al. [76] en 1974 et graphes planaire par Hopcroft et Wong en 1974 [77]. Malgré leur pertinence historique et leur valeur scientifique liée à leur complexité polynomiale, cette famille d'algorithmes d'appariement de graphes est restreinte à n'être utilisée que dans des zones applicatives spécifiques, essentiellement là où les graphes correspondants ont toujours la même structure prédéfinie.

La plupart des algorithmes de correspondance exacte travaillant sur n'importe quel type de graphe sont basés sur différentes formes de recherche d'arborescence avec suivi en arrière. L'idée de base est qu'une correspondance partielle (initialement vide) est itérativement élargie, en lui ajoutant de nouvelles paires de nœuds appariés. La paire est choisie en utilisant certaines conditions nécessaires pour assurer sa compatibilité avec les contraintes imposées par le type correspondant par rapport aux nœuds mappés jusqu'à présent, et en utilisant également certaines conditions heuristiques pour tailler dès que possible les chemins de recherche infructueuses. Cette taille est la moyenne pour augmenter l'efficacité des algorithmes d'appariement, et plus les graphes sont asymétriques, plus l'accélération est élevée. La présence d'attributs de nœuds et d'arrêtes permet généralement une amélioration impressionnante du temps d'appariement. Finalement, soit l'algorithme trouve une correspondance complète, soit il atteint un point où le mapping partiel courant ne peut pas être plus étendu en raison des contraintes. Dans ce dernier cas, l'algorithme recule, c'est-à-dire annule les dernières additions jusqu'à ce qu'il trouve une correspondance partielle pour laquelle une variante d'extension est possible. Si tous les mappages possibles qui répondent aux contraintes ont été déjà essayés, l'algorithme s'arrête. La figure 3.2 illustre la raison d'être de cette importante classe d'algorithmes : la figure 3.2.a rapporté l' appariement des graphes G1 et G2, la figure 3.2.b présente la recherche en arbre avec les solutions partielles (en termes de mappage) associées à chaque nœud de cet arbre, et la figure 3.2c la solution partielle en matière de graphes associés à l'état S8. Notons que chacun des états

intermédiaires représente une solution isomorphique partielle au problème : par exemple, S8 est la solution partielle associant respectivement les nœuds 1, 2 et 3 de G1 aux nœuds A, B et C de G2 ; il est simple de vérifier que ces sous-graphes de G1 et G2 sont isomorphes.

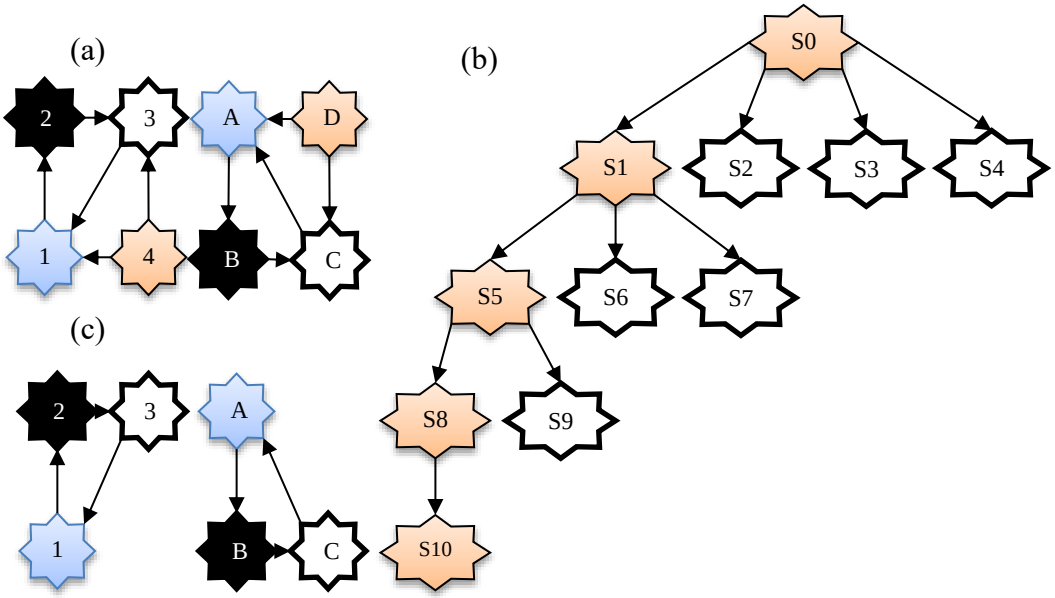


Figure 3-2. Exemple de recherche d'arborescence avec suivi en arrière

Plusieurs algorithmes attribués à cette catégorie ont été proposés ; ils diffèrent dans l'ordre dans lequel les correspondances partielles sont visitées. Probablement, le plus simple est la recherche de la profondeur, qui exige moins de mémoire que d'autres et se prête très bien à une formulation récursive. Une propriété intéressante de tels algorithmes est qu'ils peuvent être très facilement adaptés pour prendre en compte les attributs des nœuds et des arrêtes pour contraindre l'appariement désiré, sans limitation du type d'attributs pouvant être utilisés. Ceci est très important pour les applications de reconnaissance des formes où souvent les attributs jouent un rôle clé dans la réduction du temps de calcul de l'appariement.

Le premier algorithme important de cette famille est dû à Ullmann, en 1976 [78], encore largement utilisé aujourd'hui. En outre, l'approche proposée par Schmidt et Druffel en 1976 [79] adopte la même stratégie, avec l'addition d'un prétraitement qui crée une partition initiale des nœuds du graphe sur la base de la matrice de distance, pour réduire l'espace de recherche. Ghahraman et al. [80] ont proposés un autre algorithme de monomorphisme intéressant basé sur le back tracking en 1980. Ils ont manipulés l'espace de recherche en utilisant un soi-disant Netgraphe obtenu à partir du produit cartésien des nœuds de deux

graphes étant appariés. Les monomorphismes entre ces deux graphes correspondent à des sous-graphes particuliers du Netgraphe. Un inconvénient majeur de l'algorithme est que le Netgraphe est représenté en utilisant une matrice de taille $N^2 \times N^2$, où N est le nombre de nœuds du plus grand graphe. Par conséquent, seuls les petits graphes peuvent être raisonnablement traités.

Un algorithme plus récent à la fois pour l'isomorphisme et l'isomorphisme sous graphique est l'algorithme VF, proposé par Cordela et al. En 1998 [81]. Les auteurs définissent une heuristique basée sur l'analyse des ensembles de nœuds adjacents à ceux déjà considérés dans le mappage partiel. Cette heuristique est rapide à calculer, conduisant dans de nombreux cas à une amélioration significative par rapport à Ullmann et d'autres algorithmes, comme montré dans [82]. En 2004, les auteurs proposent une modification de l'algorithme [83], appelé VF2, qui réduit l'exigence de mémoire de $O(N^2)$ à $O(N)$ par rapport au nombre de nœuds dans les graphes, ce qui rend l'algorithme particulièrement intéressant pour travailler avec de grands graphes.

Une méthode récente de recherche d'arbres pour l'isomorphisme a été proposée par Larrosa et Valiente en 2002 [84] ; les auteurs reformulent l'isomorphisme du graphe comme un problème de satisfaction de contraintes (PSC), une approche qui a été étudiée très profondément dans le cadre de l'optimisation discrète et de la recherche opérationnelle, de façon à l'appliquer à l'heuristique d'appariement des graphes dérivée de la littérature PSC. Dans un article de 2011, Ullmann [85] présente une amélioration substantielle de son algorithme d'isomorphisme sous graphique bien connu depuis 1976. Le nouvel algorithme incorpore plusieurs idées de la littérature sur le problème de satisfaction de contraintes binaires, dont l'isomorphisme de sous-graphe peut être considéré comme cas particulier. L'algorithme utilise une représentation compacte en matière de vecteurs de bits pour les contraintes binaires, ce qui permet d'exécuter plusieurs opérations en parallèle, améliorant ainsi la vitesse de la mise en œuvre. Aussi Zampelli et al. En 2010 [86] proposent une méthode computationnelle efficace utilisant le paradigme de contrainte de Satisfaction, basée sur une extension de la technique introduite par Larrosa et Valiente en 2002. Plus précisément, ils présentent un algorithme pour l'isomorphisme sous graphique basé sur la recherche d'arbres, avec l'utilisation d'une étape de filtrage pour élaguer les appariements possibles, sur la base du PSC. Un autre développement de la technique, avec l'introduction

d'un meilleur filtrage basé sur la contrainte « All Different », est proposé par Solnon [87] en 2010.

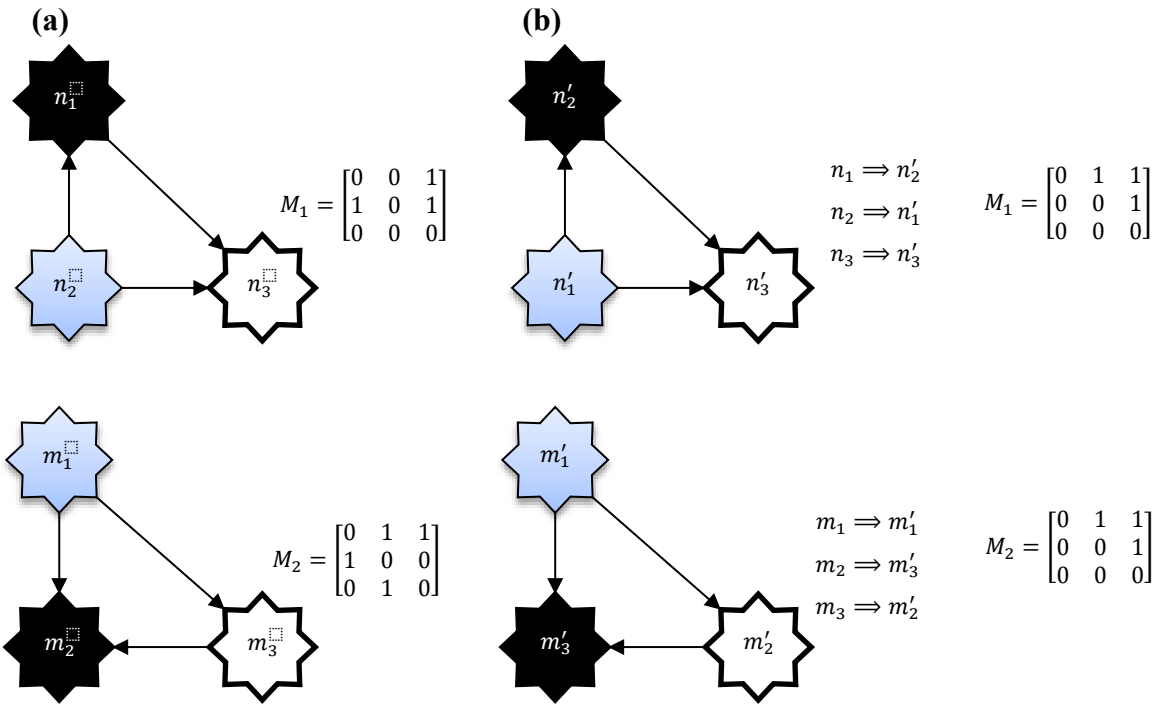


Figure 3-3. Appariement basé sur l'étiquetage canonique

Probablement, l'algorithme d'appariement le plus intéressant qui n'est pas basé sur la recherche d'arbre est Nauty, développé par McKay en 1981 [88]. Il traite uniquement le problème de l'isomorphisme et utilise certains résultats issus de la théorie des groupes pour construire le groupe automorphisme de chacun des graphes d'entrée. On en déduit un étiquetage canonique, de sorte qu'on peut vérifier l'isomorphisme de deux graphes en vérifiant simplement l'égalité de leurs formes canoniques. La logique de l'algorithme est montrée à la Figure 3.3. La vérification de l'égalité peut être effectuée en temps $O(N^2)$, mais la construction de l'étiquetage canonique peut nécessiter un temps exponentiel dans le pire des cas, alors que dans le cas moyen, cet algorithme a une très bonne performance. Cependant, cet algorithme n'est pas adapté pour exploiter les attributs de nœuds et d'arrêtes des graphes, ce qui, dans de nombreuses applications de reconnaissance des formes, peut apporter une contribution inestimable pour réduire le temps d'appariement.

De nombreux algorithmes partageant la propriété commune de réduire les contraintes de temps correspondantes : certains si généraux pour traiter en même temps avec de nombreux

types de correspondances, comme le monomorphisme, l'isomorphisme, l'isomorphisme sous graphe ou le sous-graphe commun maximum ; d'autres ne font face qu'à un sous-ensemble d'entre eux. La généralité peut également être référée au type de graphiques d'entrée, à la fois en matière de topologie et la nature des attributs nœud/arrête. On s'attend à ce que, plus un algorithme est général, moins son efficacité de calcul, mais des exceptions à cette règle peuvent être trouvée.

3.2.1.2 Appariement inexact des graphes

Dans de nombreuses applications de reconnaissance des formes, les graphes observés sont sujets à des déformations dues à plusieurs causes : variabilité intrinsèque des motifs, bruit dans le processus d'acquisition et présence d'instabilités dans les étapes de traitement menant à la représentation graphique. Ayant des graphes réels qui diffèrent quelque peu de leurs modèles idéaux. Le processus d'appariement doit donc tenir compte de ces différences en affectant un coût qui désigne le degré de dis-similarité. L'algorithme doit donc trouver un mappage qui minimise le coût d'appariement.

Les algorithmes d'appariement inexact optimal sont nés quelques années plus tard après les algorithmes exacts et leur développement est en cours. Ils trouvent une solution qui est le minimum global du coût d'appariement, ce qui implique que si une solution exacte (c'est-à-dire une solution ayant un coût égal à zéro) existe, elle sera trouvée. Par conséquent, ils peuvent être considérés comme une généralisation d'algorithmes de correspondance exacte. Les algorithmes optimaux font face aux problèmes de la variabilité des graphes et ne fournissent pas nécessairement une amélioration du temps de calcul, généralement assez élevé que celui des algorithmes exacts. Des algorithmes d'appariement approximatifs ou sous optimaux, garantissent seulement un minimal local du coût d'appariement, proche du global. Même s'il existe une solution exacte, ils ne sont peut-être pas capables de la trouver et cela est généralement inacceptable; de toute façon, la sous optimalité est abondamment remboursé par un temps de correspondance généralement polynomial.

3.2.1.2.1 Méthodes d'appariement inexact optimal

La plupart des algorithmes d'appariement de graphes inexacts, sont basés sur la définition d'un coût d'appariement obtenu comme la somme des coûts (choisis empiriquement) affectée à un ensemble d'opérations d'édition de graphes. Par exemple, le coût d'insertion de nœud est égale

à 2, le coût de la suppression d'arrête est 1, le coût d'ajout d'arrête est 1 et le coût de changement d'étiquette est de 1. La séquence d'opérations les moins chers nécessaires pour transformer l'un des deux graphes dans l'autre est calculés et son coût est défini comme la distance d'édition du graphe. Figure 3.4 illustre ce concept.

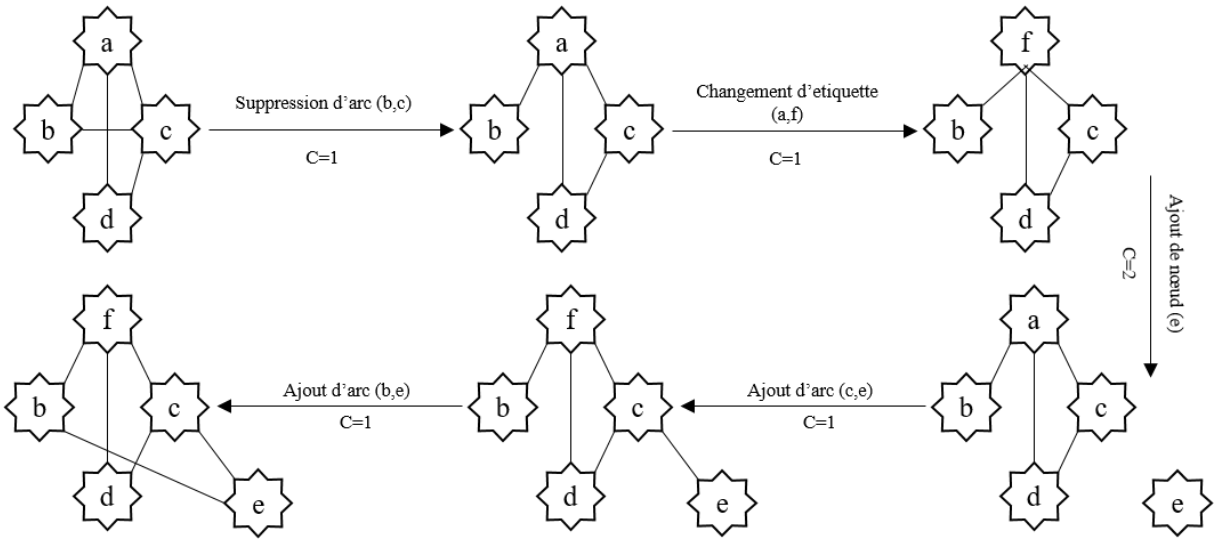


Figure 3-4. Minimum séquences d'édition (égale á 6) pour transformer G_1 en G_2

Comme pour les algorithmes d'appariement exact, la recherche de la séquence d'édition de coût minimum est généralement effectuée en utilisant une recherche d'arbre avec retour arrière. Dans ce cas, la recherche est habituellement dirigée par le coût de l'appariement partiel obtenu jusqu'à présent et par une estimation heuristique du coût d'appariement pour les nœuds restants. Ces informations peuvent être utilisées pour déterminer l'ordre dans lequel l'arbre de recherche doit être parcouru, comme dans l'algorithme [89]. Dans ce dernier cas, si l'heuristique fournit une estimation étroite du coût d'appariement futur, l'algorithme trouve la solution assez rapidement; mais si ce n'est pas le cas, l'exigence de mémoire est considérablement plus grande. Le premier algorithme inexact basé sur l'arbre est dû à Tsai et Fu, en 1979 [90], et dans une version étendue en 1983 [91]. L'article introduit une définition formelle de la correction d'erreur de l'association de graphes relationnels attribués (AGR) basés sur l'introduction d'un coût d'édition de graphes et définit une méthode de recherche garantissant la solution optimale. Un article plus récent de Wong et al. [92] en 1990 propose une amélioration de l'heuristique de Tsai et Fu pour le monomorphisme de correction d'erreur, en prenant également en compte le coût futur de l'appariement d'arrêtes. Une approche similaire est utilisée par Sanfeliu et Fu en 1993 [93], où la définition d'une vraie distance

d'édition graphique est tentée, et un algorithme sous optimal, fonctionnant dans un temps polynomial, est introduit.

Riesen et Bunke apportent des contributions pertinentes à l'évaluation de la distance d'édition du graphe en 2009 [94], où un algorithme efficace pour obtenir une distance d'édition de graphes sous optimale, basé sur l'appariement de graphes bipartites est introduit. Les méthodes permettent à l'utilisateur, au prix d'accepter une solution sous optimale, d'obtenir le résultat en temps polynomial. Cette réalisation est particulièrement pertinente du point de vue applicatif, car elle permet de comparer des graphes de plus grandes tailles en temps acceptable.

Cordella et al. [95] ont proposé un autre algorithme inexact en 1998. Ils traitent des déformations en définissant un modèle de transformation dans lequel, dans des conditions appropriées, un sous-graphe peut être effondré en un seul nœud. Le modèle de transformation est contextuel, dans le sens où une transformation donnée peut être sélectivement autorisée en fonction des attributs des nœuds voisins et des arrêtes.

D'autres approches, dans la même ligne, mais qui ne définit pas exactement une séquence d'édition, sont celles de Serratosa et al. En 1999 [96, 97], qui présentent une méthode d'appariement inexact qui exploite également une certaine forme d'information contextuelle. Les auteurs définissent une distance entre les graphes fonctionnels (DFG) qui sont des graphes relationnels attribués (ARG) enrichis d'informations supplémentaires relatives à la probabilité conjointe des nœuds afin de modéliser avec une DFG un ensemble d’ARG observés.

3.2.1.2.2 Appariement inexacte sous-optimal

Les méthodes d'appariement examinées jusqu'ici reposent sur une formulation du problème d'appariement directement en matière de graphes. Une approche radicalement différente consiste à transformer la correspondance graphique, qui est en soi un problème d'optimisation discrète, en un problème d'optimisation non linéaire continue en supprimant certaines contraintes. Cela nous permet d'utiliser l'un des nombreux algorithmes d'optimisation continue. La correspondance dans l'espace des graphes devient ainsi équivalente à la minimisation (maximisation) d'une fonction continue, correctement obtenue à partir des graphes d'entrée. L'optimisation est généralement réalisée par des améliorations successives

jusqu'à ce qu'un minimum (maximum) stable de cette fonction soit atteint. Une fois qu'une solution a été trouvée, il est nécessaire de la reconvertir du domaine continu dans l'espace graphique initial selon un processus qui peut introduire un niveau supplémentaire d'approximation. Cette approche est motivée par deux points de force : la possibilité d'utiliser la plupart des méthodes d'optimisation qui ont un fond mathématique bien établi et connu comme la relaxation de l'étiquetage et la maximisation d'attente (EM), et le coût de calcul extrêmement réduit, généralement polynomial et avec un exposant faible sur la taille des graphes.

Un des travaux exploitants la relaxation d'étiquetage est due à Fischler et Elschlager en 1973 [98] : l'idée de base est que chaque nœud d'un des graphes peut être assigné à un label à partir d'un ensemble discret d'étiquettes possibles, qui détermine le nœud correspondant de l'autre graphe. Au cours du processus d'appariement, il y a pour chaque nœud un vecteur des probabilités de chaque étiquette candidate, réévalué dynamiquement jusqu'à ce que le processus converge vers une solution stable. À ce stade, on choisit pour chaque nœud l'étiquette ayant la probabilité maximale. Il est intéressant de noter qu'à la fin du processus d'optimisation, chaque nœud reçoit une étiquette, mais il n'y a aucune garantie que chaque étiquette est affectée à un seul nœud. Il est donc nécessaire de vérifier la cohérence de la solution obtenue, et même si le problème d'appariement a une solution, il n'est pas nécessairement trouvé.

Une famille de méthodes différentes est basée sur la formulation du problème d'appariement de graphe en tant que problème de correspondance de graphe pondéré. Il consiste à trouver une correspondance, généralement exprimée au moyen d'une matrice correspondante M, entre un sous-ensemble des nœuds du premier graphe et un sous-ensemble des nœuds du second graphe. La cible est l'optimisation d'une fonction définie sur M de sorte que plus sa valeur est élevée, plus la somme des poids de l'arrête préservée par la correspondance est supérieure.

Parmi les premiers travaux basés sur cette formulation, il y a les travaux d'Almohamad et Duffuaa en 1993 [99], dans lesquels le problème quadratique est linéarisé et résolu en utilisant l'algorithme simplex, tandis que la solution approximative continue ainsi obtenue est ensuite convertie en forme discrète en utilisant la méthode dite « hongroise » pour le problème d'affectation. Rangarajan et Mjolsness [100] ont proposés en 1994 une méthode basée sur des réseaux de relaxation lagrangienne dans laquelle les contraintes sur les lignes et sur les

colonnes de la matrice correspondante sont satisfaites séparément et ensuite assimilées par un multiplicateur de Lagrange. Dans un article publié en 1996, Gold et Rangarajan [101] présentent l'algorithme de correspondance de graphes d'affectations graduées utilisant une technique connue sous le nom de non-convexité graduée pour éviter les faibles optima locaux.

Une autre classe importante d'algorithmes inexacts d'adaptation de graphe fait face au problème d'exploitation des propriétés spectrales des graphes : les valeurs propres et les vecteurs propres de la matrice d'adjacence d'un graphe sont invariants par rapport aux permutations de nœud. Par conséquent, si deux graphes sont isomorphes, leurs matrices d'adjacence auront les mêmes valeurs propres et même vecteurs propres. Malheureusement, l'inverse n'est pas vrai : on ne peut déduire de l'égalité des valeurs propres / vecteurs propres que deux graphes sont isomorphes. Cependant, puisque le calcul des valeurs propres / vecteurs propres est un problème bien étudié, qui peut être résolu en temps polynomial, il y a un grand intérêt dans leur utilisation pour l'appariement des graphes.

Le travail pionnier sur les méthodes spectrales est le document d'Umeyama, en 1998 [102], qui propose un algorithme pour l'isomorphisme pondéré entre deux graphes. Elle utilise la décomposition en éléments propres des matrices d'adjacence des graphes pour dériver une expression simple de la matrice orthogonale qui optimise la fonction objective sous l'hypothèse que les graphes sont isomorphes. À partir de cette expression, il dérive une méthode pour calculer la matrice de permutation optimale lorsque les deux graphes sont isomorphes et une matrice de permutation sous optimale si les graphes sont presque isomorphes.

En 2001, Carcassoni et Hancock [103] proposent une méthode spectrale basée sur l'utilisation de caractéristiques spectrales pour définir des partitions de nœuds susceptibles d'être appariées dans la correspondance optimale. La méthode utilise l'appariement hiérarchique en recherchant d'abord une correspondance entre les partitions et ensuite entre les nœuds dans celles-ci. Une autre méthode qui combine une approche spectrale avec l'idée de groupement a été présentée par Kosinov et Caelli en 2002 [104]: un espace vectoriel, appelé le « espace propre des graphes », est définie en utilisant les vecteurs propres des matrices d'adjacence et les nœuds sont projetés sur des points dans cet espace et un algorithme de regroupement est utilisé pour trouver des nœuds des deux graphes qui doivent être mis en correspondance.

3.2.2 Approche impure

La période impure est caractérisée par la présence de méthodes qui tentent de réutiliser les principaux algorithmes d'apprentissage et de classification, définis dans la reconnaissance statistique de formes fonctionnant dans un espace vectoriel. À cette préoccupation, il convient de rappeler le rôle exceptionnel joué par la distance d'édition, présentée à la section 3.2.1.2, dont l'introduction peut être considérée comme la date de naissance de cette période, qui s'étend principalement du milieu des années 80 jusqu'à maintenant. Avec une définition correcte des coûts d'édition et l'utilisation d'algorithmes exacts pour son évaluation, la distance d'édition du graphe donne la garantie d'être une métrique sur les graphes. Comme cela se produit normalement dans un espace vectoriel, il s'agit d'un point théorique pertinent, car il permet d'utiliser la distance d'édition comme un outil pour mesurer la similarité des objets.

3.2.2.1 Méthode de Plus Proche Voisin en graphes

La disponibilité d'une distance d'édition graphique permet de généraliser les graphes dans des méthodes de classification qui ne nécessite pas une phase d'apprentissage, tels que la classification via la méthode du plus proche voisin (NN) et ses dérivations comme la méthode de K-plus proche voisin (K-NN). Nous rappelons que ces classificateurs nécessitent simplement un ensemble de n échantillons dont l'identité est connue (ensemble de référence). La classification d'un modèle inconnu est réalisée en évaluant sa distance à partir de chaque échantillon de l'ensemble de référence et en lui attribuant la classe de l'échantillon le plus proche. Les schémas structuraux décrits par des graphes peuvent être classés en utilisant la distance d'édition de graphe (au lieu de la distance euclidienne utilisée dans les espaces vectoriels). L'algorithme 1 dans la figure 3.5 ci-dessous, montre l'extension du classificateur NN aux graphes : il est important de dire que, malgré son intelligence, qui permet de mettre en place un classificateur de graphe en un temps très court, le temps nécessaire pour effectuer la classification est généralement très élevée. L'évaluation de la distance d'édition entre deux graphes nécessite en moyenne un temps exponentiel par rapport à la taille du graphe, car le graphe d'entrée doit être comparé à chaque graphique de l'ensemble de référence.

Algorithme 1. Généralisation de la méthode NN en graphes

Entrée : *Set des graphes étiquetés* $\{(G_1, C_1), \dots, (G_n, C_n)\}$ *et l'inconnue graphe* G_x

Sortie : Étiquète C_x *du graphe* G_x

Début

$d_{min} \coloneqq +\infty$;

$j \coloneqq 0$;

for $i \coloneqq 1$ ***to*** n ***do begin***

$d_{ix} \coloneqq DistanceEdition(G_x, G_i)$;

if $d_{ix} < d_{min}$ ***then begin***

$j \coloneqq i$;

$d_{min} = d_{ix}$;

end

end;

$C_x = C_j$

end

Figure 3-5 Généralisation de la méthode NN en graphes

3.2.2.2 Méthode de K- means en graphes

Le classificateur NN des graphes est le résultat fondateur des méthodes impures. En suivant les mêmes lignes directrices, d'autres réalisations importantes ont été atteintes : c'est le cas des algorithmes de regroupement bien connu, opérant dans un espace vectoriel. Ils permettent à l'utilisateur d'obtenir le prototype d'un ensemble d'échantillons, représenté comme des points dans l'espace vectoriel, comme le centre de gravité. Cette définition simple requiert la disponibilité d'une distance à utiliser pour l'évaluer :

Avec n points : $P^k = \left(P_1^k, P_2^k, \dots, P_m^k\right)$ (avec k = 1, n) dans un espace m-dimensionnel le centre de gravité $\bar{S}$ peut être obtenu en calculant simplement la moyenne de chaque coordonnée unique des points considérés :

$$\bar{S} = (\bar{S}_1, \dots, \bar{S}_i, \dots, \bar{S}_m); \quad \bar{S}_i = \frac{1}{n}\sum_{k=1}^{n} P_i^k \qquad (3.3)$$

Ce qui précède ne peut pas être simplement étendu aux graphes, car on ne peut pas introduire les opérations d'addition entre les graphes et la multiplication d'un graphe par une constante. Le problème est surmonté en remplaçant le concept du graphe moyen d'un ensemble de graphes $\bar{S}$ par celui du graphe médian $\hat{S}$ du même ensemble S à l'aide de la distance d'édition graphique au lieu de la distance euclidienne. L'extension est définie par Jiang et al. Dans [105]:

$$\hat{S} = arg \min_{G_i \in S} \sum_{G_j \in S} D_g(G_i, G_j) \qquad (3.4)$$

D_g étant la distance d'édition entre deux graphes.

En regardant l'équation 3.2, il est clair que le graphe médian $\hat{S}$ d'un ensemble donné S de graphes, coïncide avec l'un de ces graphes. Cette définition est affectée par un problème significatif : si les membres de S sont peu nombreux et éloignés l'un de l'autre, il se peut que le graphe médian obtenu soit sensiblement différent de ce qu'on aimerait obtenir en tant que représentant du cluster.

Ce problème a été résolu par une extension adaptée et simple de la définition précédente: le graphe médian généralisé $\hat{S}$ d'un ensemble donné S est basé sur la création, à partir de l'ensemble S, d'un autre ensemble U contenant un nombre beaucoup plus élevé de graphes: U est obtenu en construisant tous les graphes possibles en utilisant les graphes non étiquetés pouvant être obtenus à partir de S et en leur attribuant toutes les combinaisons possibles d'étiquettes de nœuds et d'arcs (extraites des graphes en S). Il est trivial de réaliser que « U » augmente exponentiellement sa taille par rapport à S. L'application à l'ensemble obtenu, « U », de la définition présentée dans l'équation 3.2 rend possible la définition du graphe médian généralisé, « $\bar{S}$ », d'un ensemble S comme suit :

$$\bar{S} = arg \min_{G_i \in U} \sum_{G_j \in S} D_g(G_i, G_j) \qquad (3.5)$$

En gros, la création de U est un moyen d'augmenter le nombre de graphes de S ; de cette façon, la distance de $\bar{S}$ par rapport au centre idéal est beaucoup plus faible que celle de $\hat{S}$. En dehors de la complexité informatique élevée de l'évaluation de U à partir de S et de la nécessité de calculer beaucoup de distances d'édition de graphes (chacune exige un temps exponentiel dans le pire des cas), nous avons atteint une autre grande mesure : l'extension des

graphes au k-means, probablement l'une des méthodes les plus utilisées dans la reconnaissance des formes statistiques [106].

3.2.2.3 Quantification des vecteurs d'apprentissage en graphes

Un paradigme d'apprentissage supervisé important dans la reconnaissance statistique des formes est la quantification vectorielle d'apprentissage (LVQ). Il est un précurseur des cartes auto-organisatrices (SOM) et permet d'obtenir, avec un ensemble d'apprentissage marqué S, les prototypes représentant S au mieux. LVQ a d'abord été défini dans un espace vectoriel à la distance euclidienne. Ayant réalisé que tout algorithme nécessitant la distance entre les points dans l'espace vectoriel peut être converti en un équivalent fonctionnant sur des graphes en substituant la distance euclidienne par la distance d'édition graphique, on peut essayer d'obtenir l'équivalent du LVQ pour les graphes dans l'algorithme 2 dans la figure 3.6 ci-dessous. Par souci de simplicité de la notation, cette version fait référence au cas de n classes et à un nombre de prototypes égal à n ; Son extension à un certain nombre de prototypes $m \geq n$ est simple. Malgré sa simplicité apparente, l'algorithme 2 cache un problème dans le portage de LVQ aux graphes : la fonction MoveGraphTo, qui correspond à une opération triviale dans un espace vectoriel, n'a pas d'équivalent direct pour les graphes.

Étant donné un motif d'entré x, le prototype provisoire P_k qui est le plus proche de x est déterminé de façon préliminaire. Si la classe du prototype provisoire est correcte et identique au motif d'entré x, le prototype est mis à jour dans P_k' en le déplaçant vers x dans la direction de la différence entre x et P_k, par une fraction α de cette différence: $\Delta = \alpha\,(x - P_k)$. De même, si la classe du prototype provisoire est fausse, le prototype est déplacé de x dans la direction opposée en utilisant une fraction β différente, donc ayant $\Delta = -\beta(x - P_k)$.

Notons que dans le LVQ pour les espaces vectoriels, Δ est une somme pondérée de vecteurs; comme il s'agit de graphes, le calcul de la somme pondérée entre graphes est donc nécessaire. Ainsi, la mise à jour du prototype provisoire du graphe nécessite sa transformation dans un autre graphe plus semblable à G_x (également G_x est un graphe) par une fraction α de leur distance. Cette opération a été définie simple et intelligemment dans [107], en partant du calcul de la distance d'édition entre P_k et G_x. Nous rappelons que ce dernier donne la séquence (moins onéreuse) des opérations d'édition $e_1, e_2, \ldots, e_k$ ayant les coûts $c_1, c_2, \ldots, c_k$ qui transforment P_k en G_x en passant par les graphes intermédiaires $G_1, G_2, \ldots, G_k$. Ainsi, si on

effectue seulement un nombre d'opérations d'édition (disons r), dont les coûts s'élèvent à une fraction α de la distance d'édition totale, on peut considérer le graphe G_r comme celui qui est plus proche de G_x , par rapport à P_k, par la valeur α selon les besoins.

Algorithme 2. Généralisation de la méthode LVQ en graphes

Entrée : *Set des graphes étiquetés* $\{(G_1, C_1), \dots, (G_n, C_n)\}$ *et l'inconnue graphe* G_x

Sortie : *le centre de gravité des k classes* $\{R_1, \dots, R_k\}$

begin

$R_1, \dots, R_k \coloneqq k$ *graphes choisis aléatoirement de* $G_1, \dots, G_n$

répéter

$i \coloneqq$ *a indice aléatoire de* $1, \dots, n$

$w \coloneqq argmin_j DistanceEdition(G_i, R_j)$

si $w \coloneqq C_i$ ***alors***

$R_w \coloneqq MoveGraphto(R_w, G_i, \alpha)$

sinon

$R_w \coloneqq MoveGraphto(R_w, G_i, -\beta)$

$update\ \alpha\ and\ \beta$

Jusqu'une solution stable trouvée ou maximum itération atteinte

end

Figure 3-6 Généralisation de la méthode LVQ en graphes

3.2.2.4 Graphes noyaux

Une autre façon d'étendre les algorithmes vectoriels aux graphes se fait à travers les graphes noyaux, dont l'exploitation peut être datée de la fin des années 90, est actuellement considérée comme un sujet de recherche ouvert. Ils permettent un résultat pratique, représenté par la possibilité d'utiliser des graphes dans les machines de noyau bien connues, telles que les

Machines Vecteur Support (SVM), le perceptron multicouche (MLP) et l'analyse des composants principaux (PCA).

La pertinence théorique et pratique des machines noyaux dans les espaces vectoriels est aujourd'hui considérée comme incontestable: sous les hypothèses du théorème de Mercer, il est possible de mapper implicitement un espace vectoriel S sur un espace produit interne V en utilisant la valeur du noyau calculée sur deux points dans l'espace d'origine comme un substitut du produit intérieur des points correspondants dans l'espace V. Le concept est illustré à la figure 3.7 : donné un espace vectoriel $\mathbb{S}$ avec une fonction noyau K, il existe un espace vectoriel (pas nécessairement connu) V, ayant généralement une plus grande dimensionnalité, et une fonction M appliquant S sur V, telle que :

$$K(V_i, V_j) = M(V_i).M(V_j); \qquad \forall V_i, V_j \in S \tag{3.6}$$

Supposant que K est définie symétrique et positive.

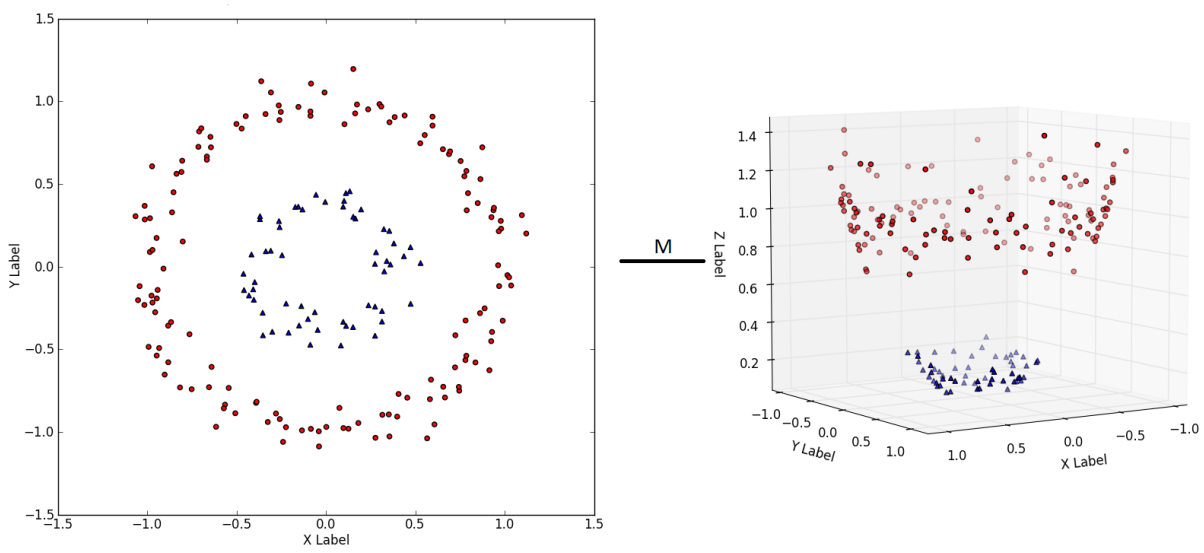


Figure 3-7. Transformation d'un problème non linéairement séparable en un problème linéairement séparable via l'application d'une fonction noyau M

Les implications de l'astuce du noyau sont énormes : si le problème initial n'est pas linéairement séparable dans S, il peut devenir linéaire dans l'espace transformé (inconnu) V. de plus, le produit intérieur entre vecteurs dans l'espace transformé (qui représente une mesure de similarité entre Les objets correspondants) est remplacé par l'évaluation du noyau.

L'extension de l'utilisation des machines du noyau aux graphes peut être faite en définissant des fonctions de noyau adaptées sur l'ensemble des graphes $\mathbb{G}$:

$$K: \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{R} \qquad (3.7)$$

La fonction K doit satisfaire la propriété de symétrie, c'est-à-dire que pour tout couple de graphes de $\mathbb{G}$, il est vrai que $K(G_1, G_2) = K(G_2, G_1)$; de plus elle doit être positive définie, c'est-à-dire pour tous les n-tuples des graphes $G_1, \dots, G_n$ et pour tous les n-tuples de valeurs réelles $c_1, \dots, c_n$, on a :

$$\sum_{j=1}^{n} \sum_{i=1}^{n} c_i . c_j . K(G_i, G_j) \geq 0; \quad \forall G_1, \dots, G_n \in \mathbb{G}, \quad \forall c_1, \dots, c_n \in \mathbb{R} \qquad (3.8)$$

Il est intéressant de noter que le théorème de Mercer ne tient pas pour les graphes, ainsi que ses conséquences. Bien sûr, un noyau adapté devrait être en mesure d'attraper la similitude entre les graphes (le bon pour l'application à la main). Un exemple de graphe noyau est $K(G_1, G_2) = n_1 . n_2$. Il est immédiat de réaliser qu'on ne peut pas attendre de bons résultats avec ce noyau: le produit du nombre de nœuds entre les graphes ne peut pas être considéré comme une bonne mesure de similarité entre eux.

L'efficacité d'un noyau doit être assurée par le fait que, pour tout ensemble de couples de graphes appartenant à G, si la valeur du noyau sur un couple est supérieure à celle de l'autre, le premier couple est constitué de graphes plus similaires que celles du deuxième, S étant une fonction de similarité définie sur l'espace du graphe.

Une classe importante de noyaux est donnée par les noyaux de diffusion, définis sur un ensemble T de n graphes et toute mesure de similarité de graphe s. A partir de la matrice de similarité par paires $S_{i,j} = s(G_i, G_j)$ il est possible de définir les noyaux de graphe comme diffusions de S. Les exemples sont le noyau exponentiel K_e et le noyau de Neumann K_n, respectivement donnés par :

$$K_e(G_i, G_j) = \sum_{k=0}^{\infty} \frac{1}{k!} \lambda^k S^k ; \qquad K_n(G_i, G_j) = \sum_{k=0}^{\infty} \lambda^k S^k \qquad (3.9)$$

Notons que la dépendance de ces noyaux par rapport aux graphes d'entrée G_i et G_j est cachée dans la matrice $S(G_i, G_j)$. Il est intéressant de noter que, comme la matrice S apparaît dans l'équation 3.7 élevé à la puissance k, le noyau sur les graphes G_i et G_j stockés dans l'élément

$K_{i,j}$ dépend non seulement de ces deux graphes mais aussi des autres dans T. La similarité utilisée comme base pour la diffusion noyau peut être dérivée par la distance d'édition, comme proposé dans [108].

Les noyaux de convolution déduisent la similitude des objets structurés à partir de la similitude de leurs parties, et c'est la raison pour laquelle ce type de noyau apparaît comme particulièrement adapté pour traiter des descriptions structurelles, naturellement faites de parties plus simples interconnectées. En particulier, à partir des similitudes entre les différentes parties, une opération de convolution est appliquée pour les transformer en une fonction noyau. Le calcul de ce noyau nécessite une décomposition préliminaire d'un graphe dans l'ensemble de toutes ses décompositions. Supposons qu'une fonction $D(G)$ décompose le graphe G en un ensemble de composantes de tuples d-dimension $D(G) = \{(g_1, \dots, g_d) \dots\}$. Par exemple, $D(G)$ pourrait décomposer le graphe dans l'ensemble des paires de nœuds adjacents (dans ce cas $d = 2$).

Pour chacune des dimensions d, nous calculons le noyau $K_i(g_i^1, g_i^2)$ où g_i^1 est la i-ème composante d'une décomposition du graphe G_1. Ainsi, un noyau de convolution peut être défini comme :

$$K(G_1, G_2) = \sum_{(g_1^1,\dots,g_d^1)\epsilon D(G_1)} \sum_{(g_1^2,\dots,g_d^2)\epsilon D(G_2)} \prod_{i=1}^{d} k_i(g_i^1, g_i^2) \qquad (3.10)$$

De nombreux noyaux de convolution ont été proposés, basés sur différents types de sous structures plus simples. Par exemple, Graphlets kernels [109] sur la base du nombre de sous-graphes communs de deux graphes. Mahe et Vert [110] et Shervashidze et Borgwardt [111] ont proposés de comparer l'ensemble des sous arbres de deux graphiques. D'autres sont basées sur des parcours [112], des sentiers [113] ou des chemins.

Une approche différente consiste à définir un noyau sur la base de la distance d'édition du graphe. Alors que la distance peut également être utilisée pour les noyaux de diffusion, il existe d'autres approches qui définissent une fonction de noyau sur la base des distances avec un ensemble de graphes de référence. Dans ce cas, l'univers des graphes sur lequel le noyau est défini n'a pas besoin d'être fini et connu à l'avance comme avec les noyaux de diffusion.

Un avantage par rapport aux grains de convolution est que les noyaux basés sur cette approche ne reposent pas sur l'hypothèse considérant qu'un sac de motifs conserve la plupart des informations de son graphe associé.

La difficulté principale dans la conception de ces graphes noyaux est que la distance d'édition ne correspond généralement pas à une métrique. Ainsi, les noyaux triviaux basés sur des distances d'édition sont généralement définis non positifs. Neuhaus et Bunke [108] ont proposé plusieurs noyaux basés sur des distances d'édition. Ces noyaux sont soit basés sur une combinaison de distances d'édition de graphes (noyau trivial, noyau de graphe de zéros), soit incorporent dans les schémas de construction du noyau plusieurs caractéristiques déduites du calcul de la distance d'édition. Ce domaine de recherche est aujourd'hui très actif et les résultats sont prometteurs, même si des limitations pratiques sévères sont imposées par le fait que la plupart d'entre elles exigent que l'ensemble des graphes soit connus a priori.

3.2.3 Approche extrême

Une extrémité du processus de réutilisation des méthodes statistiques de reconnaissance des formes sur les graphes est la soi-disant incorporation de graphe (Graph Embedding) : sous cette dénomination, on catégorise toutes les méthodologies visant à représenter un graphe entier (éventuellement avec des attributs attachés à ses nœuds et arrêtes) comme un point dans un espace vectoriel approprié. Bien sûr il existe d'innombrables façons de mapper un graphe à un vecteur; Mais les intégrations intéressantes se caractérisent par le fait qu'ils préservent la similarité des graphes: plus deux graphes sont similaires, plus les points correspondants sont dans l'espace vectoriel ciblé. Bien que des ouvrages séminaires sur ce domaine soient déjà présents dans la littérature antérieure, c'est dans la dernière décennie que ces techniques ont gagné en popularité dans la communauté des relations publiques. Bunke et Riesen [46] présentent une révision sur les graphes noyaux et l'incorporation des graphes, et dans [47] ils étendent l'examen présentant ces techniques comme un moyen d'unifier les méthodes statistiques et structurelles en reconnaissance des formes.

L'intégration de graphes peut être considérée comme un pont réel reliant les deux mondes: une fois que l'objet à la main a été décrit en matière de graphes, et le dernier représenté dans l'espace vectoriel, tous les problèmes d'appariement, d'apprentissage et de regroupement peuvent être résolus via les méthodes statistiques de reconnaissance des formes. Les

approches possibles proposées pour effectuer l'incorporation du graphe incluent des méthodes spectrales [114, 115], basées sur la décomposition propre de la matrice d'adjacence ou d'une matrice qui lui est associée. Dans [116], une technique statistique connue sous le nom d'échelle multidimensionnelle (MDS) est utilisée pour intégrer un graphe, caractérisé par la matrice des distances géodésiques entre nœuds, dans un collecteur. Dans [117], un algorithme d'intégration pour un cas particulier des arbres est proposé. L'algorithme est basé sur le calcul du super arbre commun minimum d'un ensemble d'arbres. Chaque arbre est alors représenté comme un vecteur dont les éléments encodent les nœuds du super arbre qui se trouvent dans l'arbre. Dans [118] des marches aléatoires sont utilisées pour dériver un graphe intégré; en particulier l'encodage encode le temps attendu nécessaire pour qu'une marche aléatoire se déplace entre deux nœuds. Dans [119], l'enrobage est construit en choisissant au hasard un petit ensemble de graphes comme prototypes, et en représentant un graphe avec le vecteur des distances d'édition de chaque prototype. Puisque le calcul de distance d'édition de graphes est un problème NP-complet, une approximation de cette mesure est réellement utilisée.

3.3 Conclusion

Le présent chapitre a permis d’avoir une idée sur la façon dont les graphes ont été utilisés dans la reconnaissance des formes. En effet, dans un premier temps, la mise en correspondance des graphes a été effectuée en les projetant dans l’espace graphique, soit de façon exacte ou inexacte. Dans un deuxième temps et à l’aide de l’introduction de la distance d’édition, des méthodes statistiques de classification, groupement et apprentissage (tel que K-NN, K-moyennes et LVQ), ainsi que des machines noyaux (tel que SVM, ACP et MLP) ont été adapté pour les graphes. Ceci a représenté le premier pas vers l’unification des méthodes structurelles et statistique, En effet une solution qui permet de projeter les graphes dans l’espace vectoriel a été introduite sous forme d’une incorporation de graphes. Cependant, le processus d’adaptation des méthodes statistiques aux graphes, que ce soit à l’aide des graphes noyaux ou l’incorporation des graphes, nécessite beaucoup de calculs. Une approche qui est intéressante, qui est remédie aux problèmes de temps de calculs, est celle d’appariement spectral des graphes utilisant les propriétés spectrale pour calculer la similarité.

Ce chapitre a marqué la fin de l’étude bibliographique. Dans le chapitre suivant nous allons présenter notre contribution sur la reconnaissance des formes par les graphes, précisément la reconnaissance optique des caractères Tifinagh.

DEUXIÈME PARTIE : CONTRIBUTION

4 Reconnaissance hors ligne des caractères Tifinaghs imprimés

4.1 Introduction

Depuis la codification du caractère Tifinagh, la reconnaissance de la forme imprimée de ce caractère a fait l'objet de nombreux travaux de recherche. Ces travaux ont abouti à la réalisation de plusieurs systèmes de reconnaissance basés sur deux approches descriptives de caractères : l'approche descriptive statistique et l'approche descriptive géométrique.

Pour améliorer le taux de reconnaissance de ces systèmes, une combinaison adéquate des descripteurs et/ou une hybridation appropriée des classificateurs s'est avérée nécessaire mais au détriment du temps de reconnaissance qui devient plus prohibitif. La réalisation d'un système de reconnaissance des caractères Tifinagh précis et rapide demeure l'objectif escompté des travaux de recherche. Afin d'atteindre cet objectif, nous nous sommes proposés d'élaborer un système de reconnaissance des caractères Tifinaghs imprimés, basé sur une approche qui se distingue des approches adoptés par les travaux réalisés dans ce contexte, à savoir l'approche structurelle qui consiste à représenter les caractères par les graphes. Cette approche s'intéresse aux motifs des caractères sous forme d'un ensemble de primitives liés entre eux sans négliger leurs positions dans la forme originale.

4.2 Reconnaissance des caractères Tifinaghs imprimés par les matrices d'incidences

4.2.1 Description du système de reconnaissance élaboré

Dans cette section, nous décrivons les différentes méthodes de prétraitement, de description et de classification des images caractères, intégrées dans le système de reconnaissance que nous avons élaboré. La figure 4.1 ci-dessous présente la structure de ce système.

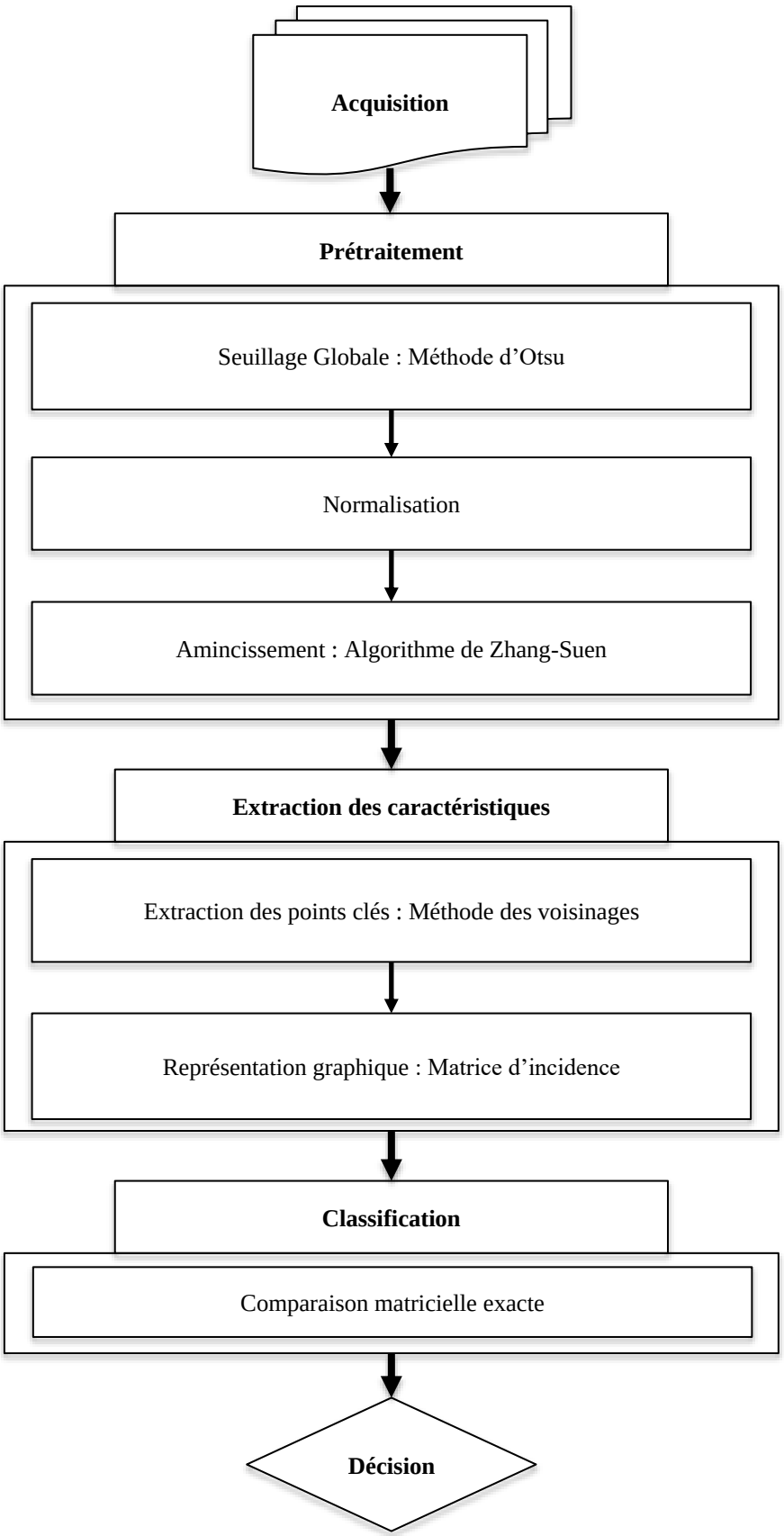


Figure 4-1. Système élaboré pour la reconnaissance des caractères Tifinaghs imprimés

4.2.2 Prétraitement

Comme on l'a déjà mentionné dans la section 1.3.1, le prétraitement vise à produire une image nettoyé en éliminant le bruit et en gardant que les informations pertinente et nécessaire pour les phases qui suivent. L'application des fonctions de prétraitement est effectuée de façon séquentielle.

4.2.2.1 Seuillage d'Otsu

Dans un premier temps, un seuillage est appliqué à l'image acquise pour produire une image binaire. Nous avons utilisé la méthode bien connue d'Otsu [4]. C'est une méthode de seuillage globale largement utilisé dans le domaine de traitement de l'image. Elle est considérée comme l'une des meilleures méthodes de seuillage (voir section 1.3.1.1). Le principe de la méthode implique l'itération à travers toutes les valeurs de seuil possibles et le calcul d'une mesure de propagation pour les niveaux de pixel de chaque côté du seuil, c'est-à-dire les pixels qui tombent soit au premier plan, soit à l'arrière-plan. Le but est de trouver la valeur de seuil où la somme de la propagation de l'avant et arrière-plan est minimale. La figure 4.2 ci-dessous présente l'image monochrome du caractère yaṛ obtenue par l'application de la méthode de seuillage Otsu.

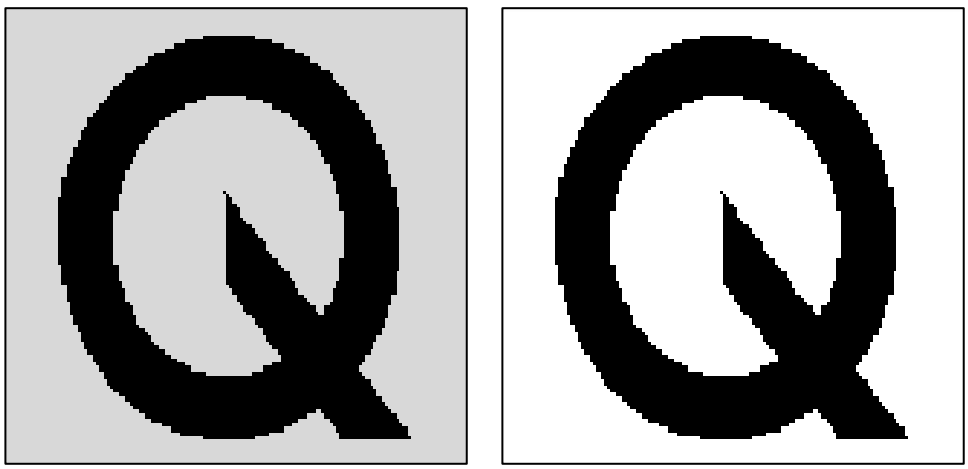


Image originale Image monochrome

Figure 4-2. Image Originale et sa version monochrome produite par le seuillage d'Otsu, exemple du caractère yaṛ

4.2.2.2 Normalisation

L'image monochrome est ensuite normalisée en supprimant le vide entre les bordures de l'image et celles du plus petit rectangle contenant le caractère. Ce rectangle est détecté en se

servant de la liste des coordonnées (x, y) des pixels de l'avant plan, tel que x est la liste des lignes et y est la liste des colonnes, en se basant sur les valeurs minimales et maximales dans la liste des lignes et colonnes. La taille $m \times n$ de l'image normalisée I est la suivante :

$$\begin{cases} m = \max\limits_{i \in x} i - \min\limits_{i \in x} i \\ n = \max\limits_{j \in x} j - \min\limits_{j \in x} j \end{cases} \quad (4.1)$$

Sachant que la valeur des pixels de l'avant plan est de 1, l'avant plan de l'image originale est inséré dans l'image normalisée de la façon suivante :

$$\forall i \in x, \forall j \in y, I(i - min\, i\,, j - min\, j) = 1 \quad (4.2)$$

La figure 4.3 ci-dessous présente l'image normalisée obtenue à partir de l'image monochrome du caractère yaṛ.

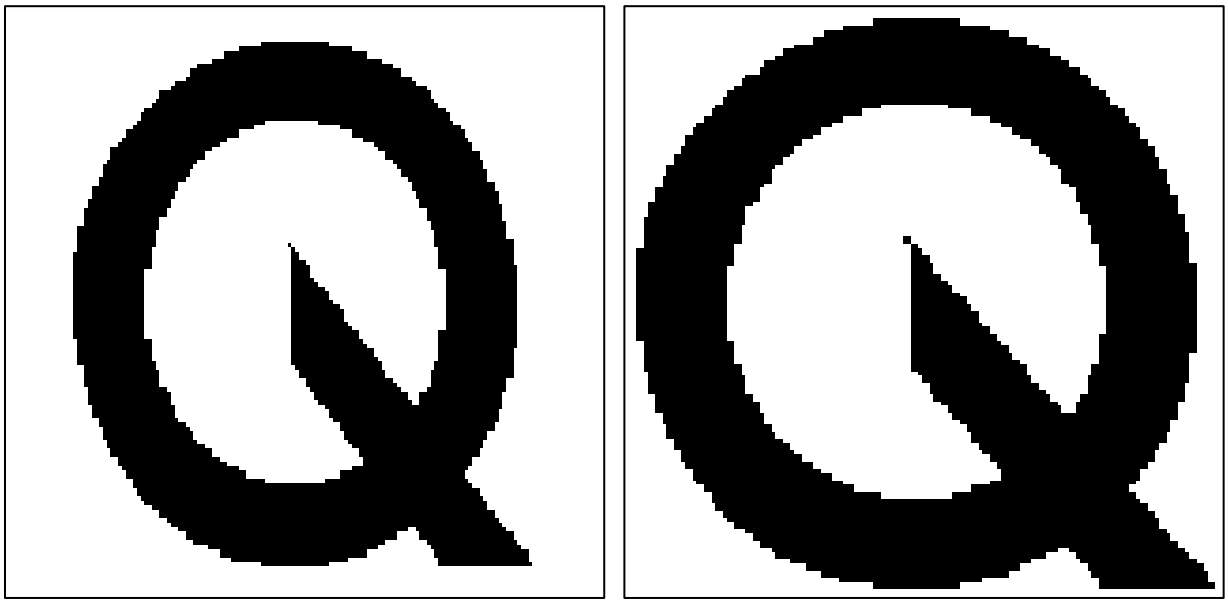


Image monochrome Image normalisée

Figure 4-3. Exemple de normalisation du caractère yaṛ

4.2.2.3 Amincissement

L'amincissement accélère considérablement toute analyse ultérieure en réduisant l'épaisseur des images des caractères. Il simplifie aussi l'extraction des propriétés géométriques telles que les points d'intérêts. Pour cela l'algorithme de Zhang-Suen [120] est appliqué à l'image normalisée du caractère. C'est un algorithme d'amincissement parallèle en deux itérations

pour éliminer les pixels qui n'appartiennent pas au squelette du caractère. Dans la première itération, un pixel $I\ (i,j)$ est supprimé si les conditions suivantes sont satisfaites :

1. Ce pixel est un pixel d'avant plan.
2. Il a au moins deux voisins d'avant plan et pas plus de six.
3. Au moins l'un des pixels suivants, est en arrière-plan : $I(i-1,j), I(i,j+1), I(i+1,j)$.
4. Au moins l'un des pixels suivants, est en arrière-plan : $I(i,j+1), I(i+1,j),\ I(i,j-1)$.

Dans la deuxième itération, les conditions 3 et 4 deviennent :

3. Au moins l'un des pixels suivants, est en arrière-plan : $I(i-1,j), I(i,j+1), I(i,j-1)$.
4. Au moins l'un des pixels suivants, est en arrière-plan : $I(i-1,j), I(i+1,j), I(i,j-1)$.

La figure 4.4 ci-dessous présente le squelette de l'image obtenue à partir de l'image normalisée du caractère yaṛ.

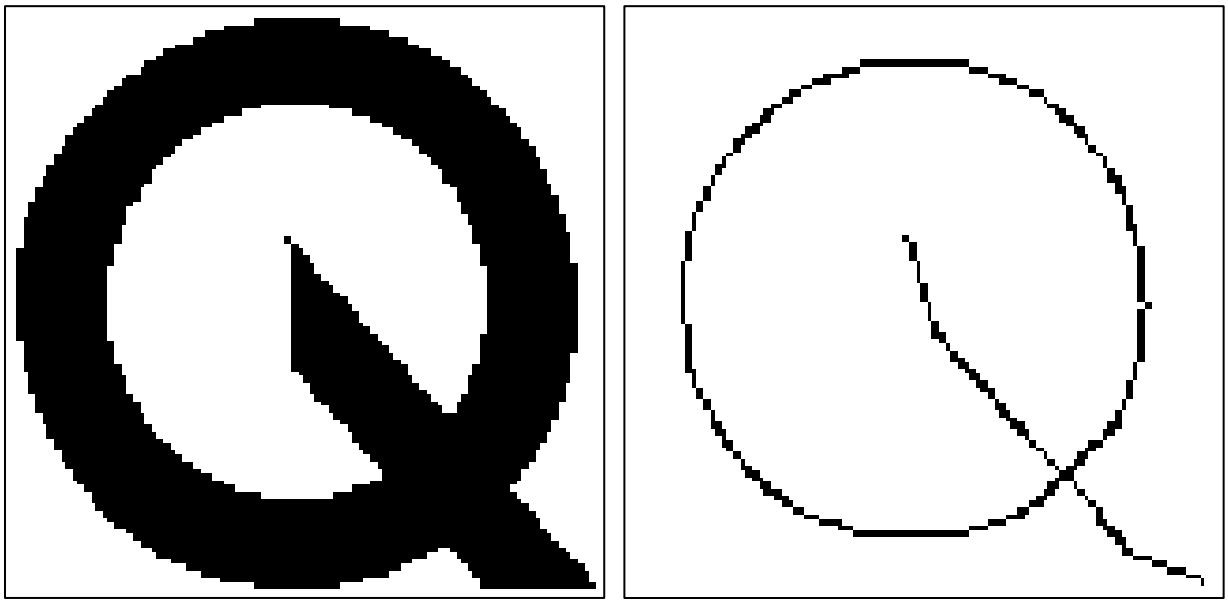


Image normalisée Image amincie

Figure 4-4. Exemple de squelettisation du caractère yaṛ

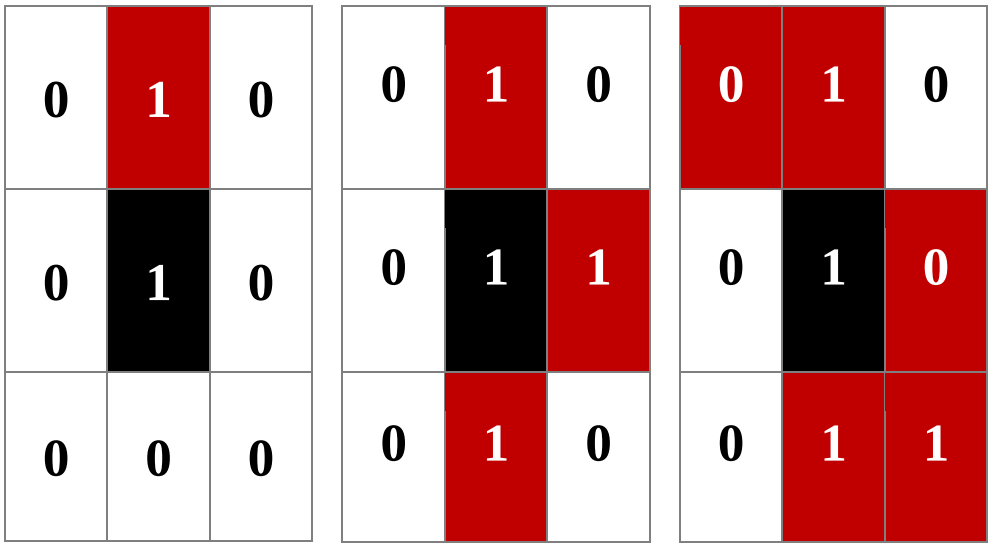
4.2.3 Extraction des primitives

Notre objectif dans cette phase est de représenter les caractères par les matrices d'incidences qui sont un moyen pour représenter les graphes (voir section 2.3.2.2). Pour atteindre cet objectif, il est nécessaire de diviser les formes des caractères en plusieurs formes géométriques simples telles que les points clés.

4.2.3.1 Extraction des points clés

Afin de représenter les caractères Tifinagh par les graphes, les points clés sont extraits en premier. Ils serviront de délimiteur pour la segmentation des caractères en segments. Dans un squelette du caractère parfait, un pixel de champ est considéré comme un point clé en comptant le nombre de pixels de l'avant plan (voir figure 4.5 ci-dessous). Si le nombre de voisins est un, alors ce pixel est considéré comme un point d'extrémité comme le montre la figure 4.5(a). Si le nombre de voisins est supérieur à deux alors ce pixel est un point d'intersection (voir figure 4.5(b)). Cependant, un squelette est rarement parfait quel que soit l'algorithme d'amincissement utilisé, on trouve souvent des parties épaisses. Pour remédier à ce problème, le nombre de transitions d'un pixel de l'arrière-plan à un pixel d'avant plan est considéré, au lieu du nombre de pixels de l'avant plan (voir figure 4.5(c)). Ce qui signifie que si le nombre de transitions est un ou plus que deux, alors nous avons un point de fin ou un point d'intersection.

Un troisième type de points est celui des points d'inflexions qui sont les points dont le nombre de transitions, dans le sens horaire, est égal à deux (voir figure 4.5(c)). La figure 4.6 ci-dessous présente un exemple d'extraction des points d'intérêt du caractère yaẓ.



(a) (b) (c)

Figure 4-5. Exemple des 8- voisinages d'un pixel d'avant plan

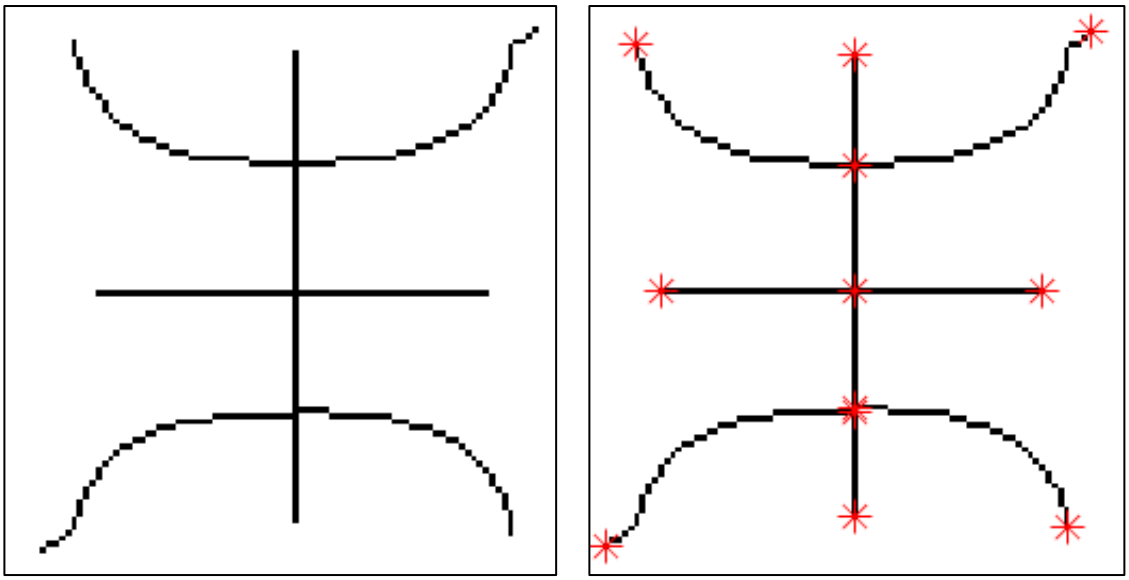


Image amincie | Image point d'intérêt

Figure 4-6. Exemple de points d'intérêt extraits pour une image du caractère yaz

L'algorithme utilisé pour l'extraction des points clé est le suivant :

Entrée : *image aminscie* « I » *du caractère*

Sortie : *listes des coordonnées des points clés* « P » *dans l'image*

tant que $i := 1$ à n ***faire***

tant que $j := 1$ à m ***faire***

$t1 := I(i-1,j) < I(i-1,j+1);$

$t2 := I(j-1,j+1) < I(i,j+1);$

$t3 := I(i,j+1) < I(i+1,j+1);$

$t4 := I(i+1,j+1) < I(i+1,j);$

$t5 := I(i+1,j) < I(i+1,j-1);$

$t6 := I(i+1,j-1) < I(i,j-1);$

$t7 := I(i,j-1) < I(i-1,j-1);$

$t8 := I(i-1,j-1) < I(i-1,j);$

$S := t1 + t2 + t3 + t4 + t5 + t6 + t7 + t8\ ;$

si $S == 1\ OU\ S > 2$ ***alors***

$P := [i,j];$

fin si

fait

4.2.3.2 Représentation graphique : matrice d'incidence

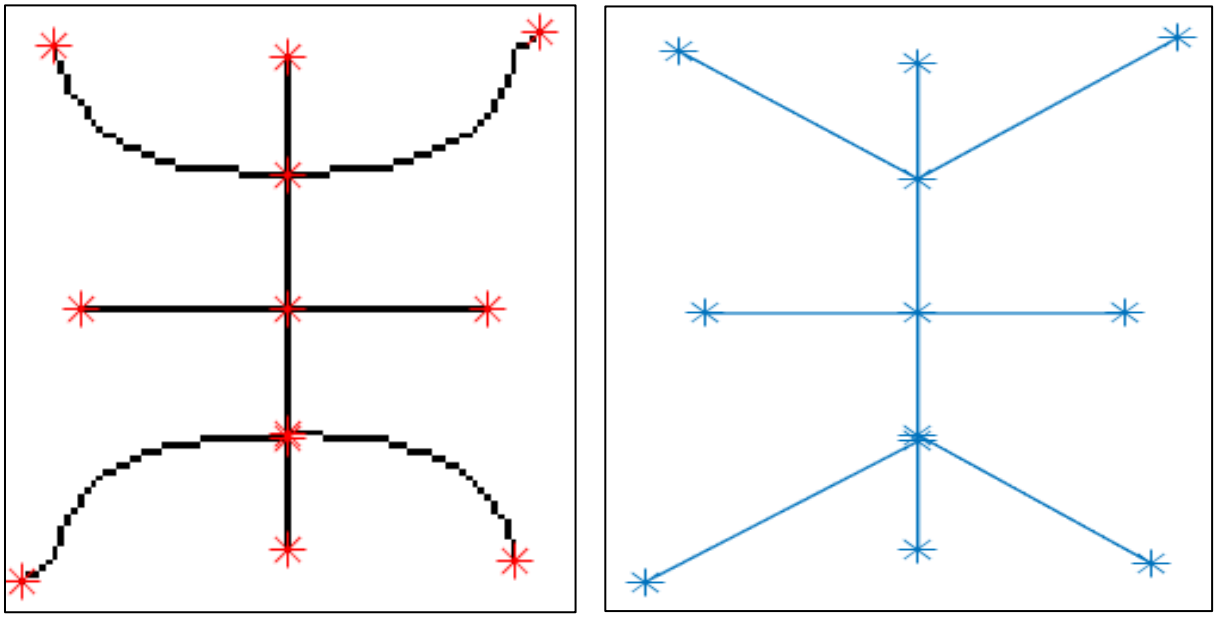
L'extraction des points clés nous permet de procéder à représenter un caractère par un graphe. Comme on l'a décrit dans la section 2.3.2.2, Un graphe est une représentation mathématique formelle d'un ensemble d'objets et de leurs relations. Chaque objet est appelé un sommet. Les relations entre les objets sont appelées arrêtes. Plus formellement, nous définissons un graphe G comme une paire ordonnée G = (V, E) où V est un ensemble de sommets, E est un ensemble d'arrêtes qui définissent la connectivité entre une paire de sommets.

La matrice d'incidence est une façon de représenter les graphes. Cette représentation est rapide et compacte. Il s'agit d'une matrice M de taille $m \times n$ où m est le nombre de nœuds et n est le nombre d'arrêtes. L'entrée dans la ligne i et la colonne j est non nul, si et seulement si le sommet i est incident à l'arrête j, ce qui signifie :

$$\begin{cases} M(i,j) = 1 \; si \; le \; sommet \; i \; est \; incident \; \text{à} \; l'arrête \; j \\ M(i,j) = 0 \; si \; non \end{cases} \qquad (4.3)$$

Après la segmentation du caractère, les caractéristiques extraites seront représentées par un graphe non orienté, où les nœuds représentent les points clés et les arrêtes représentent les segments reliant ces points par paire.

La figure 4.7 ci-dessous présente un exemple de représentation graphique du caractère yaz.



points clés Représentation graphique

Figure 4-7. Représentation graphique basée sur les points d'intérêt extraits du caractère yaz

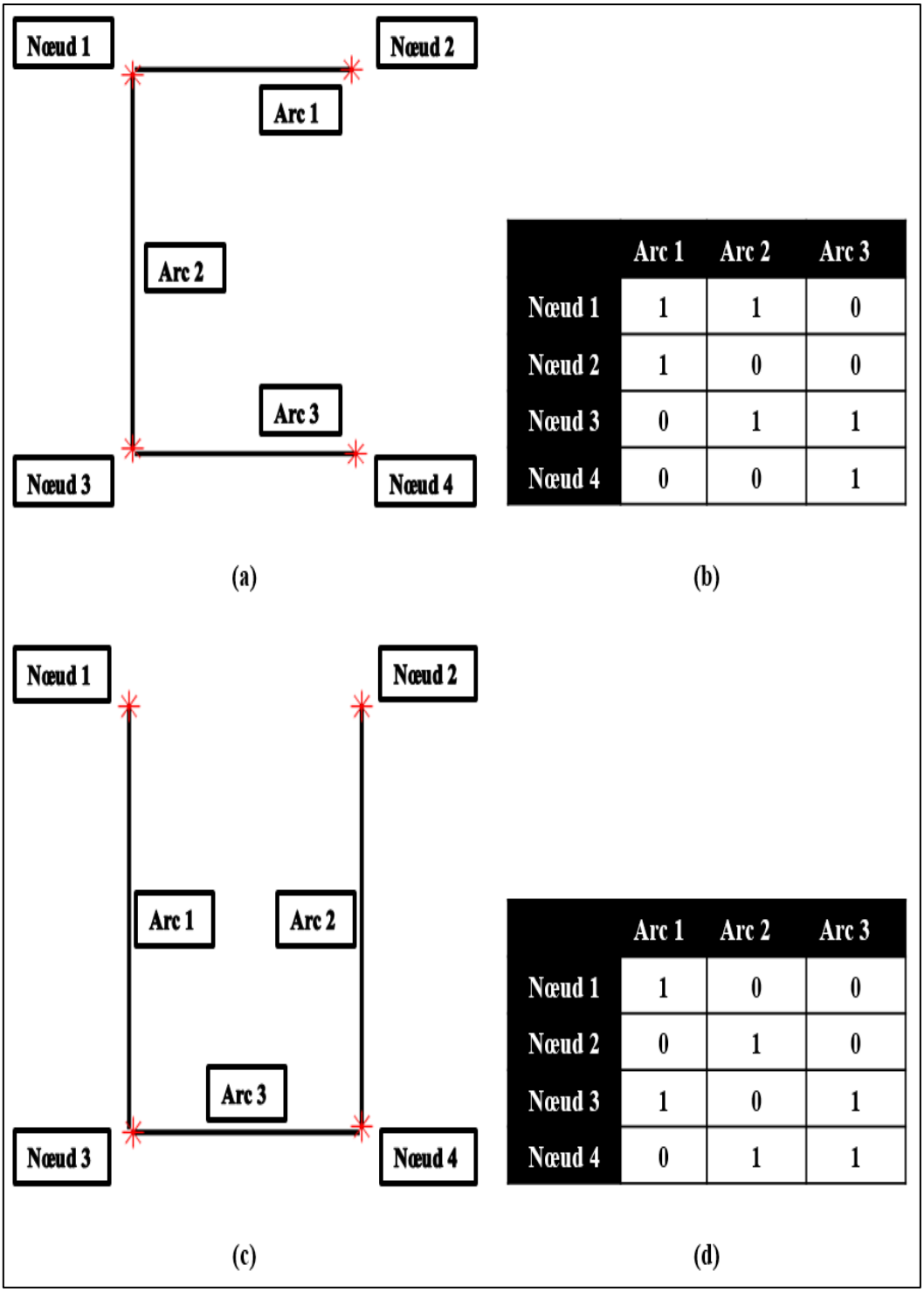
À fin de détecter les points qui sont directement connectés dans le squelette d'un caractère, nous avons commencé par les organiser selon leurs positions dans l'image en effectuant un scan de gauche à droite et de haut en bas. Les premiers points détectés sont les premiers insérés dans la nouvelle liste des points clés. Cela nous permet non seulement de simplifier la détection des points connectés mais aussi de contribuer à résoudre les problèmes d'orientation.

Pour chaque point dans la liste, l'un des points d'avant plan dans ces 8-voisinages est sélectionné. Si ce point est dans la liste des points d'intérêt, le point de départ et celle-ci sont enregistré dans une liste des points connectés. Si non, ce dernier point est marqué comme visité et les mêmes opérations sont répétées sur ceci jusqu'à atteindre un point clés. Si le point de départ n'a plus de voisinages non visité, le point suivant dans la liste des points singulier est sélectionné comme point de départ.

L'algorithme de segmentation et détection de connectivité entre les points clé procède comme suit :

1. *Sélectionner le premier point clé de la liste et marquer le comme visité* ;
2. *Sélectionner un voisinage d'avant plan* ;
3. *Vérifier si ce voisinage est un point clé* ;
4. *Si non, marquer ce dernier voisinage comme visité et appliquer l'étape* 2 *et* 3 *sur ce point* ;
5. *Si oui, marque ce dernier voisinage comme point clé connecté au point de départ* ;
6. *Appliquer l'étape* 3 à 5 à *tous les voisinages non visités du point de départ* ;
7. Sélectionner le point suivant dans la liste des points clés et revenir à l'étape 1 ;
8. *Fin de l'algorithme quand tous les points singuliers sont visités.*

Une fois que les points clés connectés sont déterminés, la matrice d'incidence est construite, en attribuant la valeur 1 aux points qui partagent le même segment. La figure 4.8 ci-dessous représente les caractères yaw et yam et leurs matrice d'incidence.



	Arc 1	Arc 2	Arc 3
Nœud 1	1	1	0
Nœud 2	1	0	0
Nœud 3	0	1	1
Nœud 4	0	0	1

	Arc 1	Arc 2	Arc 3
Nœud 1	1	0	0
Nœud 2	0	1	0
Nœud 3	1	0	1
Nœud 4	0	1	1

Figure 4-8. Exemple d'extraction de la matrice d'incidence, (a) caractère yam, (b) matrice d'incidence du caractère yam, (c) caractère yaw, (d) matrice d'incidence du caractère yaw

4.2.4 Classification

Les matrices d'incidences construites pour les différents caractères, sont utilisées pour classer les caractères Tifinaghs par une comparaison matricielle exacte entre les matrices d'incidence des images de test avec celles de référence. Cette opération est rapide lorsqu'on utilise une base de référence assez grande pour augmenter les chances de trouver une matrice d'incidence similaire dans les références. Ces matrices doivent avoir la même taille et les mêmes valeurs de leurs éléments.

4.2.5 Résultats et interprétation

4.2.5.1 Résultats

Les résultats présentés dans cette partie, sont obtenus par des expériences réalisées sur une base de données localement utilisée dans notre laboratoire. Elle est composée de 3300 images de caractères. Chaque caractère est représenté par 100 images avec différents fontes et tailles. Les tests sont effectués sous une validation croisée (les images de test ne sont pas utilisées comme références). Le tableau 4.1 ci-dessous représente les résultats obtenus en termes de taux de reconnaissance, taux d'erreur et temps d'exécution.

Tableau 4-1 Taux de reconnaissance, Taux d'erreur et Temps d'exécution obtenue

	Système élaboré
Taille des images de référence	2200
Taille des images de test	1100
Taux de reconnaissance (%)	91%
Taux d'erreur (%)	9%
Temps d'exécution (sec)	400

Tableau 4-2 Taux de reconnaissance, de confusion et de non classification obtenus

Taux en (%) / Nom du caractère	Taux de reconnaissance	Taux de confusion	Taux de non-classification
Ya	79	21	0
Yab	97	0	3
Yag	94	0	6
Yag^{w}	82	0	18
Yad	90	0	10
Yaḍ	100	0	0
Yey	91	0	9
Yef	89	0	11
Yak	100	0	0
Yak^{w}	87	0	13
Yah	94	0	6
Yaḥ	100	0	0
Yaɛ̣	95	0	5
Yax	93	0	7
Yaq	100	0	0
Yi	88	0	12
Yaj	100	0	0
Yal	83	0	17
Yam	97	3	0
Yan	94	6	0
Yu	91	0	9
Yar	91	9	0
Yaṛ	83	9	8
Yaɣ	90	0	10
Yas	79	10	11
Yaṣ	85	0	15
Yac	74	0	26
Yat	100	0	0
Yaṭ	90	0	10
Yaw	100	0	0
Yay	97	3	0
Yaz	81	0	19
Yaẓ	94	0	6

4.2.5.2 Interprétation

Bien que les résultats concernant le taux de reconnaissance et le temps d'exécution obtenus par le système élaboré soient encourageants, l'objectif escompté au niveau de la discrimination entre les caractères n'est pas atteint. En effet, l'analyse des résultats présentés

par le tableau 4.2, montre que certains caractères sont confondus avec d'autres et/ou non classés.

Une réflexion menée sur le fonctionnement des différentes parties du système élaboré, nous a permis de se rendre compte que ces effets sont engendrés par les facteurs suivants :

- La représentation du graphe par la matrice d'incidence. Dans cette représentation on ne peut pas ajouter des poids sur les nœuds ou les arrêtes pour mieux caractériser les caractères. Cette limite est la source principale de la confusion des caractères ya (ⴰ) et yar (ⵔ).
- La méthode d'amincissement adoptée. Cette méthode élémine entièrement un composant constituant le caractère comme le cercle rempli du caractère yas (ⵙ) et le segment du caractère yey (ⵄ). Cette limite a donné une confusion entre les caractères (yas, yar), (yey , yu (ⵓ)) et (yi (ⵉ), et yay (ⵢ)) ;
- L'utilisation de la comparaison matricielle exacte qui ne tolère pas la dis-similarité et nécessite une taille de la base de données assez grande. Ceci limite les performances de la classification basée sur cette technique.

Ainsi, en se basant sur ces remarques, nous avons, dans la suite de ce travail, cherché à améliorer les performances de ce premier prototype du système de reconnaissance des caractères Tifinaghs par les graphes en apportant des modifications au niveau du prétraitement, de l'extraction des primitives, de la représentation graphique et de la classification. Le système de reconnaissance ainsi amélioré « système 2 » est décrit dans la section suivante.

4.3 Système de reconnaissance amélioré « système 2 »

Dans cette partie, nous décrivons le deuxième système que nous avons réalisé pour améliorer la reconnaissance des caractères Tifinaghs, illustré par la figure 4.9 ci-dessous, en tenant compte des remarques faites précédemment. Cette amélioration est le résultat d'une amélioration de l'algorithme d'amincissement, de l'algorithme utilisé pour la détection des points d'intérêts et de l'utilisation d'une approche d'appariement inexacte des graphes pour la classification.

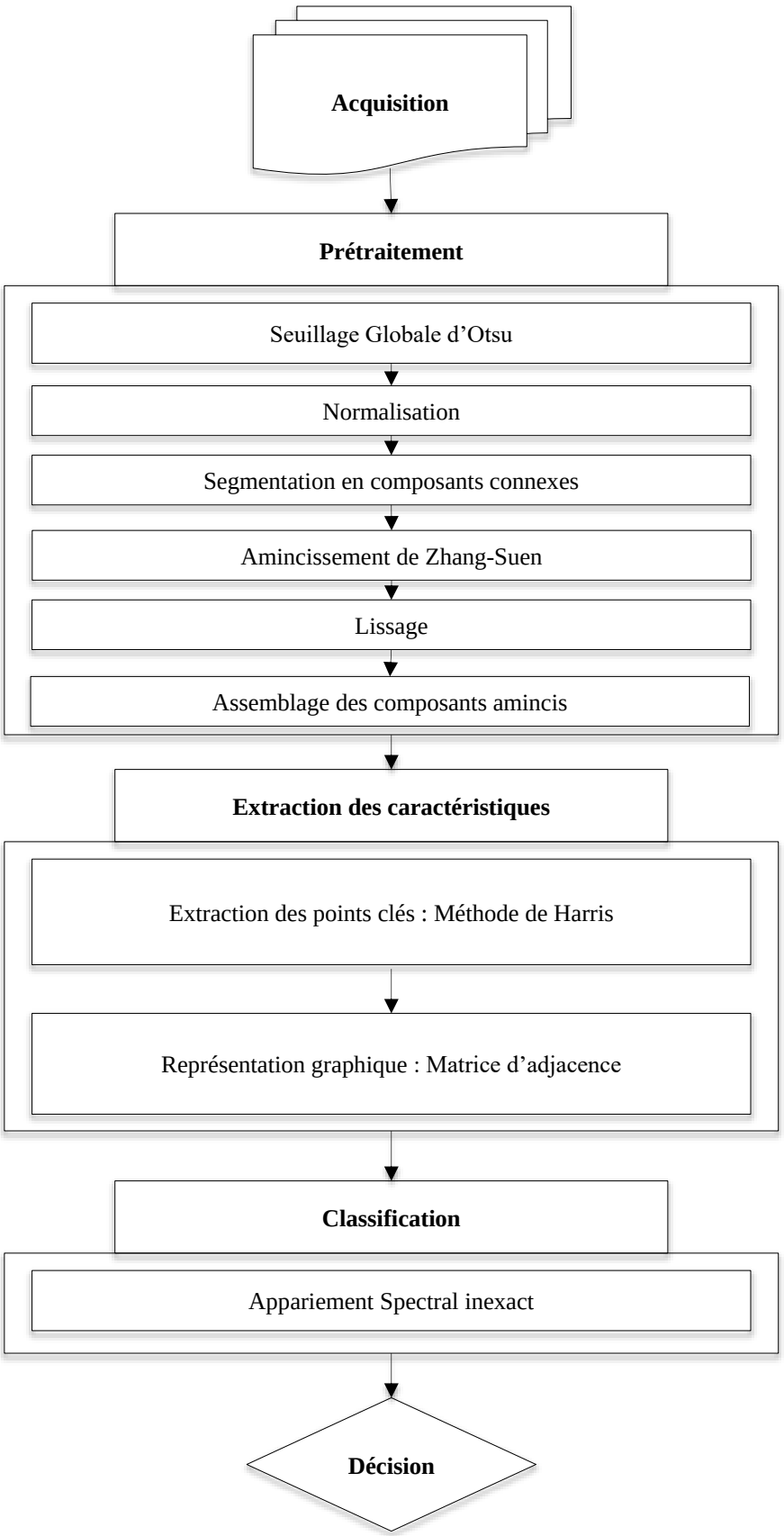
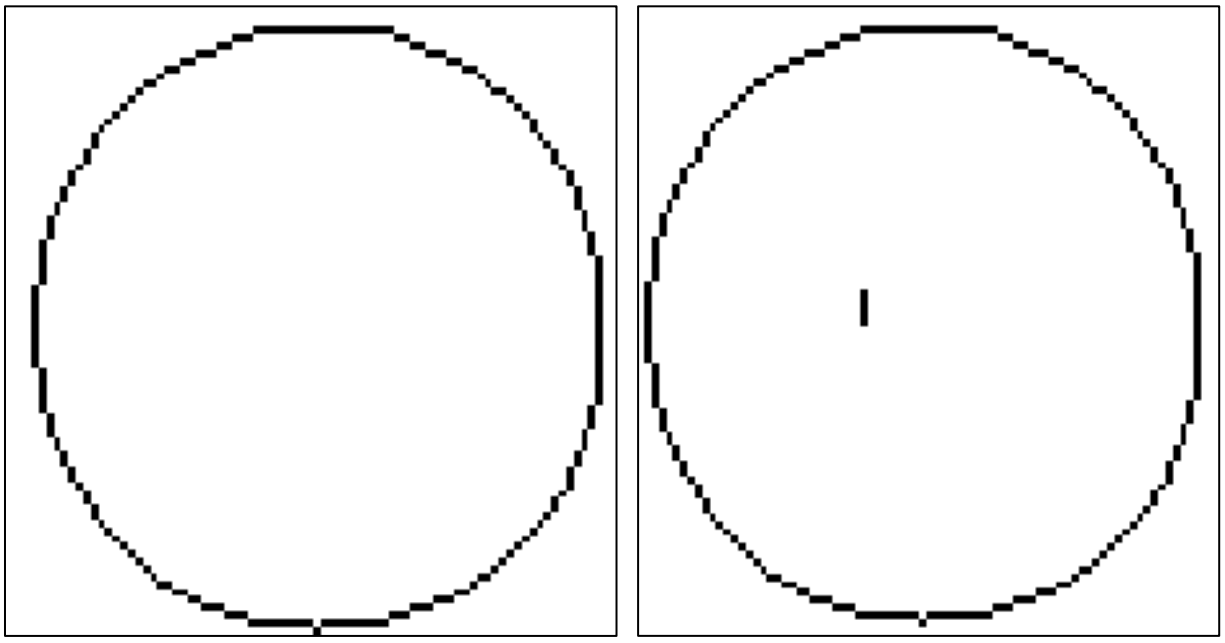


Figure 4-9. Structure du système amélioré pour la reconnaissance des caractères Tifinaghs imprimés

4.3.1 Prétraitement

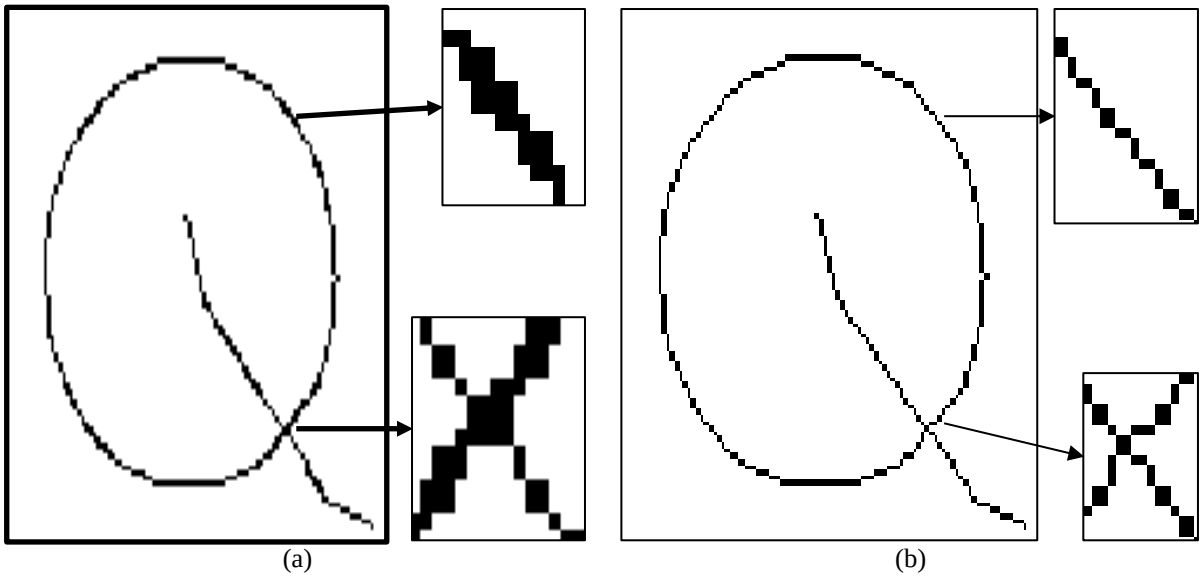
Dans cette partie nous avons modifié le processus d'amincissement. Dans un premiers temps, nous avons remarqué que dans certaines images du caractère yas (⊙), le cercle rempli est entièrement éliminé (voir figure 4.10(a)). Pour remédier à ce problème, nous avons appliqué l'algorithme de zhang-suen sur les composants connexes qui constituent le caractère au lieu de l'appliquer sur l'image entière. Cela nous a permis d'ajouter un petit trait pour remplacer les composants totalement éliminés. La figure 5.10(b) présente le résultat de cette amélioration.



(a) (b)

Figure 4-10. Exemple d'application de l'algorithme d'amincissement amélioré

Dans un deuxième temps, nous avons constaté que le squelette obtenue n'est pas correctement aminci dans les parties incurvées des caractères (voir figure 4.11(a)) ci-dessous). Pour remédier à ce problème, nous avons éliminé les pixels qui n'affectent pas la connectivité et la topologie du caractère en traitant que les pixels dont le nombre de voisinages et le nombre de transitions sont différents. La figure 4.11(b) illustre le résultat de ce lissage.



(a) (b)

Figure 4-11. (a) Problème lié à l'algorithme d'amincissement Zhang-Suen, (b) résultat après lissage

Le temps moyen de prétraitement de chaque alphabet Amazighe est illustré dans la figure 4.11.

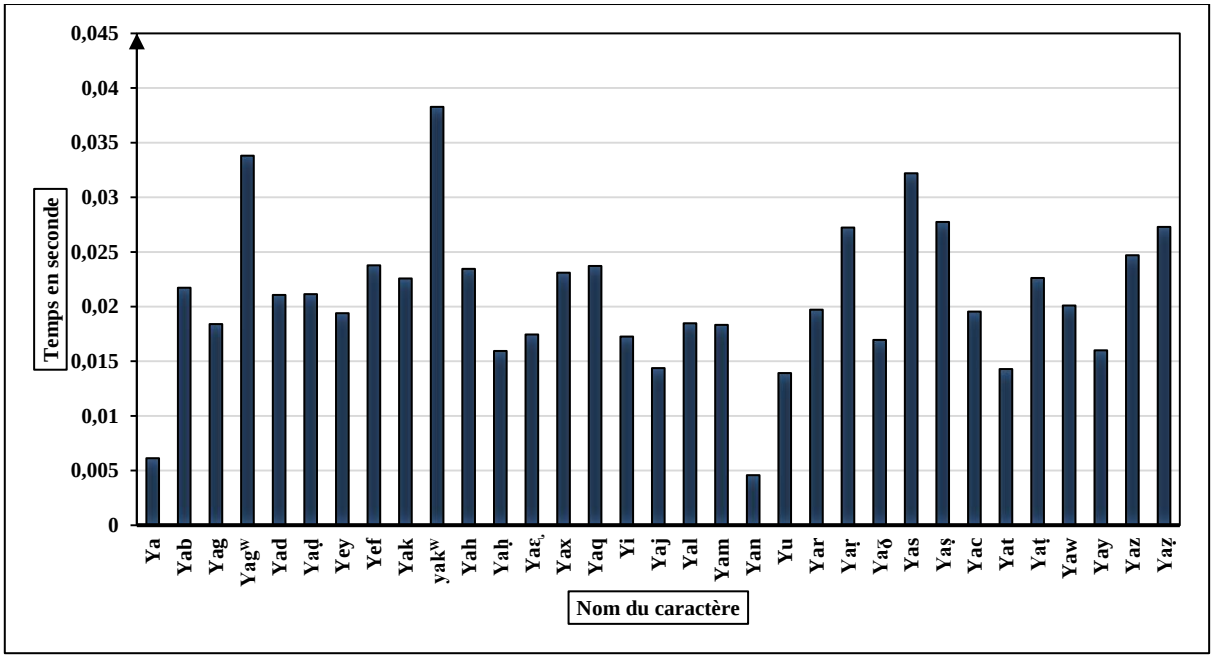


Figure 4-12. Moyenne de Temps de prétraitement de chaque caractère

Le processus d'amincissement ainsi modifié, est un processus assez rapide et permet d'améliorer l'extraction des points clés caractérisant chaque caractère.

4.3.2 Extraction des primitives : Détecteur de coins Harris

Le détecteur de coins Harris [39] est un algorithme qui permet d'effectuer une représentation discrète des caractéristiques d'une image. Son objectif est d'extraire les points d'intérêts qui donnent des informations sur la structure de l'image. C'est une amélioration du détecteur des coins Moravec.

Le principe de l'algorithme de Morvec est de considérer une fenêtre locale dans l'image pour déterminer la moyenne des changements de l'intensité dans l'image en la déplaçant vers plusieurs directions. Quand la fenêtre subit une séquence de grands changements, le minimum de ceux-ci représente le coin ou le point d'intérêt.

Si on suppose que *I* est l'intensité de l'image, le changement E produit par le déplacement (x, y) est donné par :

$$E_{x,y} = \sum_{u,v} w_{u,v} \left| I_{x+u,y+v} - I_{u,v} \right|^2 \qquad (4.4)$$

Où w est une fenêtre locale rectangulaire dans l'image qui se déplace en suivant quatre directions $(x, y) \in \{(1,0), (1,1), (0,1), (-1,1)\}$. Trouver les points d'intérêts est équivalent à la recherche du maximum locale dans $min\{E\}$.

Selon Harris, l'algorithme souffre de la limitation des mouvements et de la forme binaire et rectangulaire de la fenêtre locale ; Ainsi que du grand nombre de points qui ne sont pas vraiment des coins. L'algorithme de Harris est une solution à ces problèmes en lissant la forme de la fenêtre et en améliorant la façon dont celle-ci se déplace en prenant en compte toutes les directions possibles de déplacement. La nouvelle fonction de changement d'intensité est exprimée par l'équation 4.5 ci-dessous.

$$E(x, y) = Ax^2 + 2Cxy + By^2 \qquad (4.5)$$

avec,

$$A = \frac{\partial I}{\partial x}^2 \otimes w \qquad (4.6)$$

$$B = \frac{\partial I}{\partial y}^2 \otimes w \qquad (4.7)$$

$$C = \left(\frac{\partial I}{\partial x}\frac{\partial I}{\partial y}\right) \otimes w \quad (4.8)$$

La nouvelle fenêtre locale w est une fenêtre circulaire exprimée par la formule gaussienne suivante :

$$w_{u,v} = exp - (u^2 + v^2)/2\sigma^2 \quad (4.9)$$

Pour simplifier l'utilisation informatique de la nouvelle fonction E, on peut l'écrire sous la forme matricielle suivante :

$$E(x, y) = (x, y)M(x, y)^T \quad (4.10)$$

Tel que M est une matrice symmetric de taille 2×2 :

$$M = \begin{bmatrix} A & C \\ C & B \end{bmatrix} \quad (4.11)$$

À fin de rendre la détection des coins plus sélective, Harris a aussi introduit une mesure de qualité et de sensitivité :

$$Tr(M) = A + B \quad (4.12)$$

$$Det(M) = AB - C^2 \quad (4.13)$$

La détermination de la région de coin est effectuée par une valeur positive de la fonction de réponse R donnée par l'équation 4.14 ci-dessous.

$$R = Det(M) - kTr(M)^2 \quad (4.14)$$

L'utilisation de la méthode de Harris résout le problème de détection des points d'intérêt dans les formes circulaires. La valeur k est un seuil choisi empiriquement. Dans notre cas cette valeur est de 0.04. Le résultat de l'application cette méthode est illustré dans la figure 4.13.

Cependant, l'application de cette méthode sur des images binaires, a montré que l'extraction des points clés n'est pas satisfaisante particulièrement dans le cas des caractères qui contiennent des courbes ou des points d'intersection avec un nombre de voisin supérieur à trois. Ce problème est résolu en regroupant des points d'intérêts proches en un point tel qu'il est illustré à la figure 4.14.

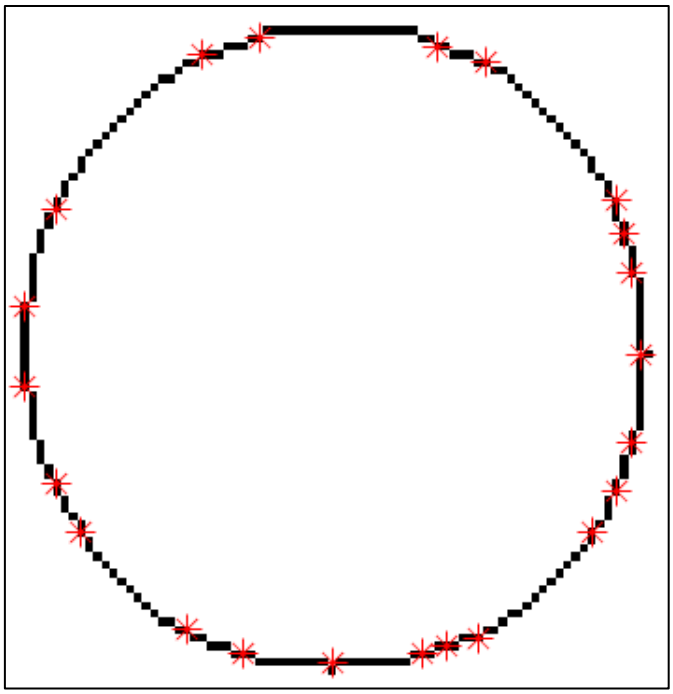
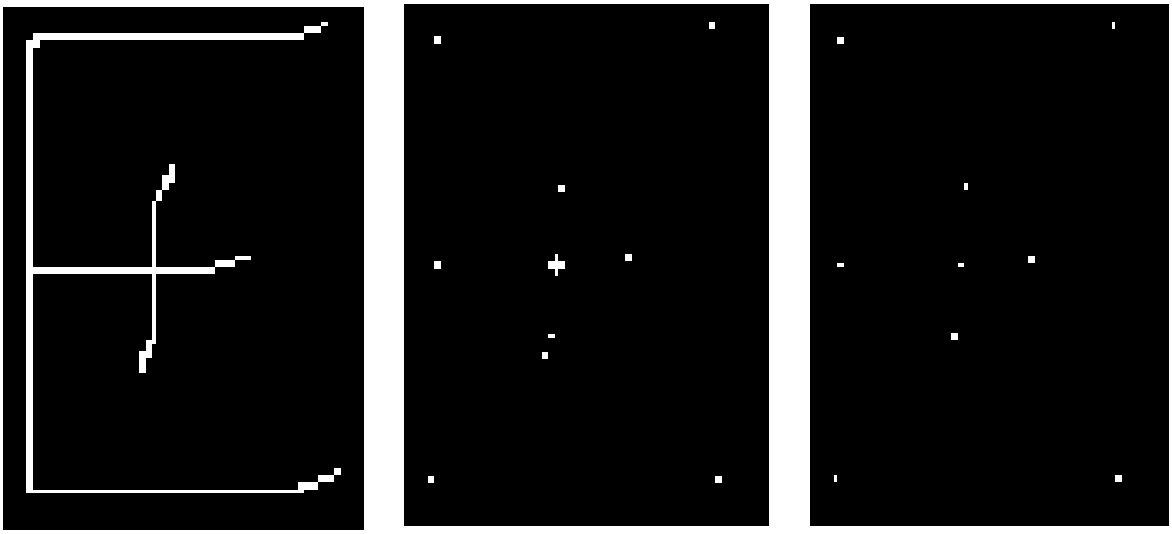


Figure 4-13. Points clés extraite à partir du squelette du caractère yar



(a) (b) (c)

Figure 4-14. Extraction amélioré des points singuliers, (a) squelette du caractère yaḍ, (b) points clés extraits par la méthode de Harris, (c) Résultat du groupement des points proches

Le temps moyen d'extraction des points clés et leur nombre pour chaque caractères sont présentés par les figures 4.15 et 4.16. Ainsi, l'utilisation de la méthode de Harris pour détecter les points clés a permis de fournir une description structurelle performante et rapide même pour les formes circulaire. Comparé aux autres algorithmes de détection de coins, la méthode de Harris est la plus robuste pour ce genre d'image. Il fait partie des algorithmes les plus utilisés pour détecter les contours et les coins.

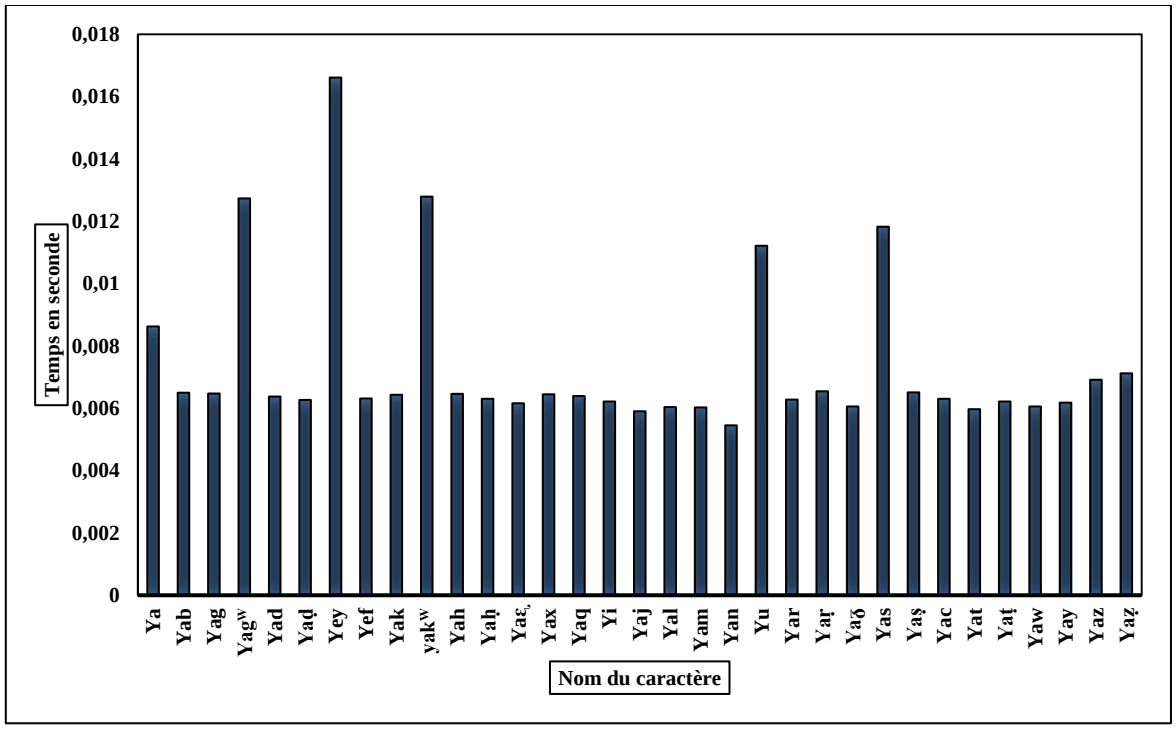


Figure 4-15. Moyenne de temps d'extraction des points singulier par la méthode de Harris

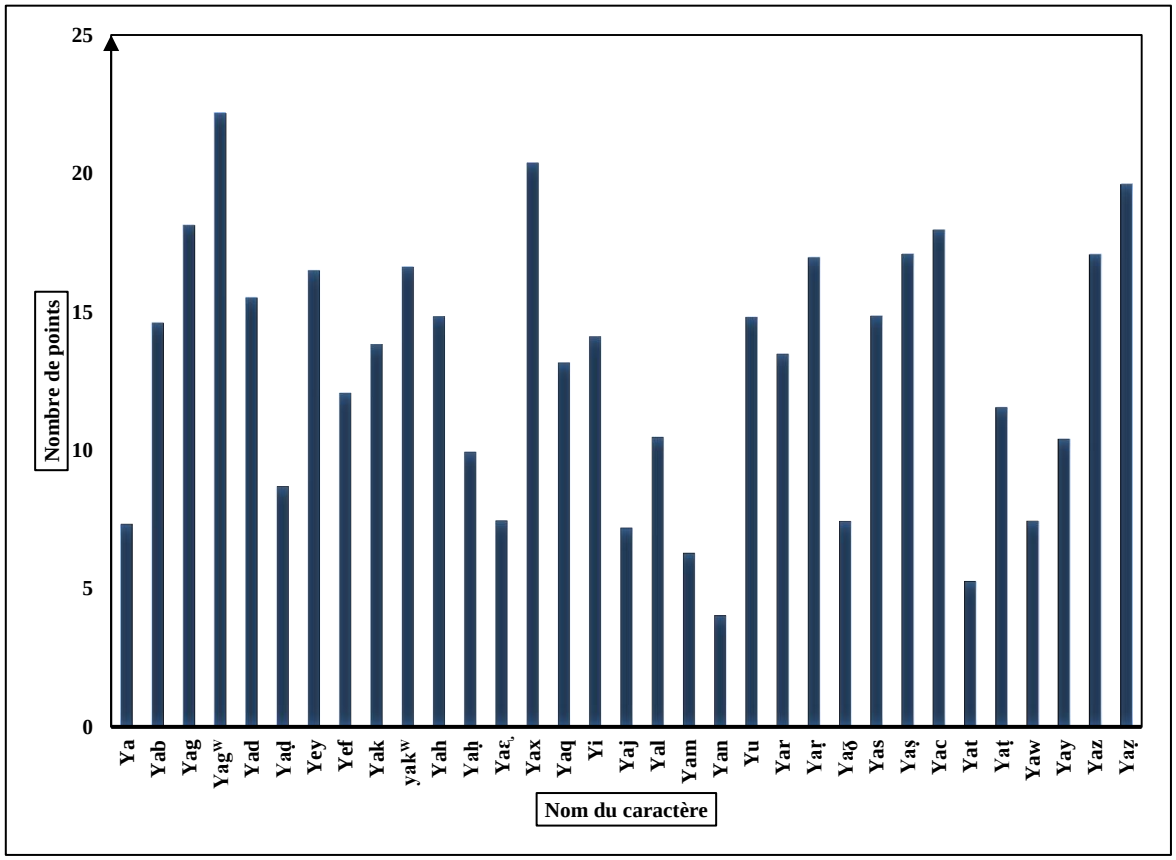


Figure 4-16. Moyenne du nombre de points extraits par la méthode de harris

4.3.3 Représentation graphique : matrice d'adjacence

La matrice d'adjacence est une façon de représenter les graphes. Il s'agit d'une matrice carrée binaire AM où le nombre de sommet | V | est sa taille. L'entrée dans la ligne i et la colonne j est non nul si et seulement si l'arrête (i, j) est dans le graphe, ce qui signifie :

$$\begin{cases} AM(i,j) \neq 0, & sommet\ i\ et\ \ j\ sont\ connectés\ par\ une\ arrête \\ AM(i,j) = 0, & sommet\ i\ et\ j\ ne\ sont\ pas\ connectés \end{cases} \quad (4.15)$$

Dans notre cas, nous avons utilisé les graphes non orientés ($AM(i,j) = AM(\mathrm{j,i})$) qui permettent une représentation graphique rapide et compacte. L'algorithme de construction graphique que nous avons mis en œuvre procède comme suit :

1. *Extraire les points clés par la méthode de Harris;*
2. *Extraire les segments reliant ces points (voir algorithme de la section 4.4.2.2);*
3. *Soit AM une matrice d'adjacence de taille* $n \times n$*. Dont n est le nombre des points clés;*
4. *Pour chaque segment, 4 informations sont extraites : premier point singulier, second points, orientation* O *(calculé à l'aide des fonctions existante dans MATLAB) du segment, longueur* L *du segment;*
5. *En se basant sur ces informations, la matrice d'adjacence est construite telle que :*

$$\begin{cases} AM(i,j) = \omega, & point\ clé\ i\ est\ connecté\ au\ point\ clé\ j \\ AM(i,j) = 0, & si\ non \end{cases} \quad (4.16)$$

Avec, $$\omega = 2 \times L + O \quad (4.17)$$

La figure 4.16 illustre un exemple de représentation matricielle du caractère Yaḍ. Les valeurs non nuls dans cette matrice représente le poids du la ligne (nœud) i avec la colonne (nœud) j.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**
1	0	0	0	0	0	0	0	0	45	0	0	0	0	0	0	0	0	0	51	0	0	0
2	0	0	0	0	0	0	0	37	0	0	0	0	0	0	0	0	0	0	0	51	0	0
3	0	0	0	0	0	0	0	37	0	0	0	19	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	17	24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	17	0	0	0	0	0	0	43	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	24	0	0	0	0	17	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	42	0	0	0	0	0	37
8	0	37	37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	45	0	0	0	0	17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	14	0	0	0	0	0	0	0	37
11	0	0	0	0	43	0	0	0	0	0	0	0	0	0	42	0	0	0	0	0	0	0
12	0	0	19	0	0	0	0	0	0	0	0	0	0	0	0	0	14	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0	14	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	14	0	0	16	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	42	0	0	0	0	0	0	0	0	0	45	0
16	0	0	0	0	0	0	42	0	0	0	0	0	0	0	0	0	0	55	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	14	14	0	0	0	47	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	55	0	2	0	0	55	0
19	51	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	56	0	0
20	0	51	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	56	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	45	0	0	55	0	0	0	0
22	0	0	0	0	0	0	37	0	0	37	0	0	0	0	0	0	0	0	0	0	0	0

Figure 4-17. Exemple de la matrice d'adjacence du caractère yar

La version binaire de cette matrice et les coordonnées des points clés dans l'image originale sont ensuite utilisés pour obtenir la représentation graphique du caractère yar, illustrée par la figure 4.17.

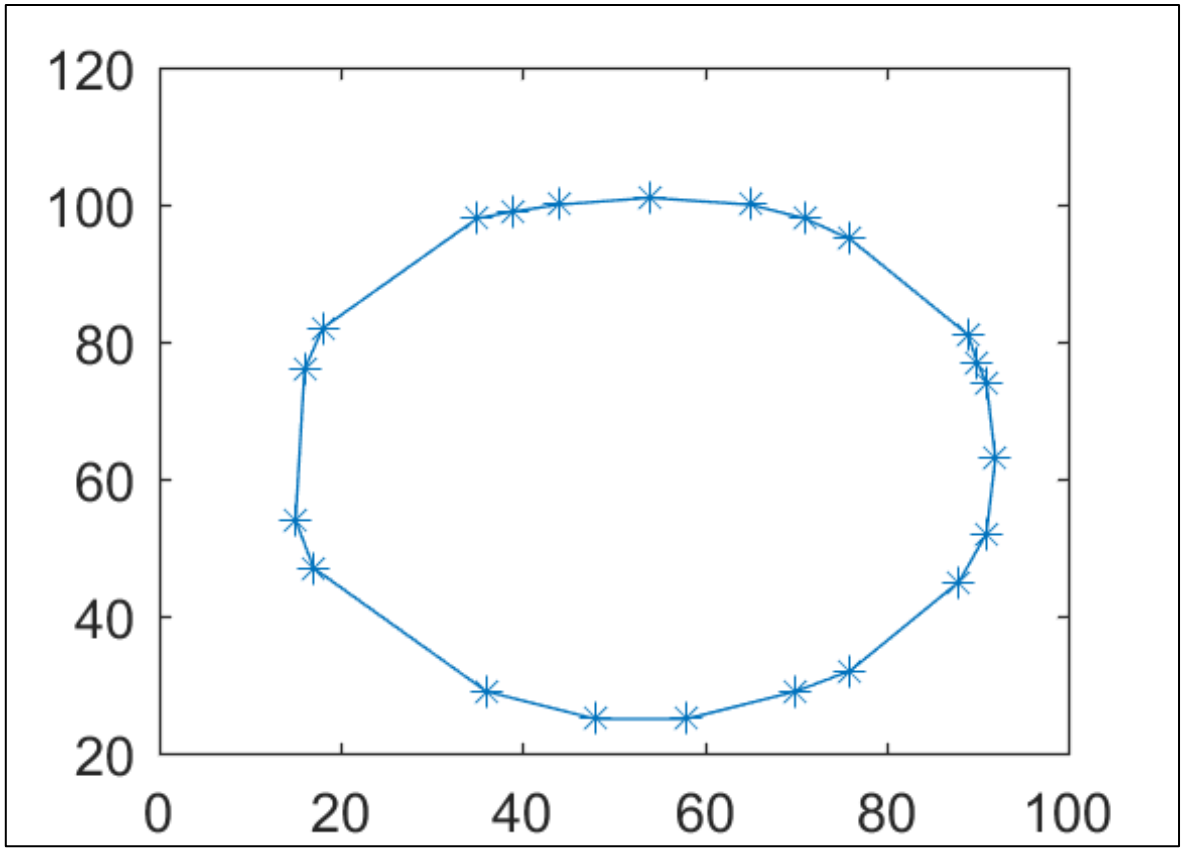


Figure 4-18. Représentation graphique du caractère yar

4.3.4 Appariement spectrale des graphes

La méthode spectrale utilisée [121] vise à trouver un accord cohérent entre deux ensembles de caractéristiques P (P contient n_P éléments) et Q (Q contient n_Q éléments). Elle est généralement utilisée pour trouver le composant principal dans un graphe à partir de la matrice de similarité M dont :

- Les lignes et les colonnes représentent les attributions potentielle d'un nœud i du graphe G_p et d'un nœud i' de G_Q ($a = (i, i'), i \in P \; et \; i' \in Q$) ;
- Les valeurs de ces éléments représentent le degré de similarité entre ces attributions, c'est-à-dire qu'une valeur non nulle signifie qu'il y a une similarité potentiel entre un nœud de G_p et un nœud de G_Q ;

- La représentation suit le mappage un-à-un de correspondance, c'est-à-dire qu'un nœud de G_p peut être similaire à un seul nœud de G_Q. Elle prend en compte la géométrie du caractère et le degré de similarité entre les segments et les points clés.

La procédure pour construire la matrice M est la suivante :

Étant donné C un ensemble de paires où $a = (i, i') \in C$. Il contient toutes les affectations possibles qui respectent le mappage par correspondance un-à-un. Pour mesurer l'accord entre les deux ensembles de caractéristiques de P et Q, une liste L est créée pour stocker le score associé pour chaque affectation candidate $a \in C$, et l'affinité associée pour chaque paire d'affectation (a, b), où $b \in C$. En se basant sur cette liste, nous mettons ces scores dans la matrice M comme suit :

- Les entrées diagonales de M contiennent le score d'affectation individuelle a;
- Les autres entrées contiennent le score de paires d'affectation (a, b).

Puisque nous avons utilisé les points clés comme caractéristiques, toutes les paires d'affectation sont des candidates. Ainsi, le problème de la correspondance est réduit à trouver le groupement C qui maximise le score inter cluster.

$$S = \sum_{a,b \in C} M(a,b) = x^T M x \qquad (4.18)$$

x est un vecteur binaire telle que :

$$\begin{cases} x(a) = 1 \; si \; a \in C \\ x(a) = 0 \; si \; a \notin C \end{cases} \qquad (4.19)$$

Pour trouver le meilleur cluster $\boldsymbol{C}^*$, nous devons trouver le vecteur binaire $\boldsymbol{x}^*$ qui maximise le score $\boldsymbol{S}$. Plus grand est le score, meilleur est la similarité entre les caractères.

$$x^* = argmax(x^T M x) \qquad (4.20)$$

Nous avons appliqué la méthode décrite ci-dessus en suivant l'algorithme suivant :

1. *Construire la matrice symétrique et positive M ;*
2. *Initialiser L par toutes les affectations possibles et x comme un vecteur nul ;*
3. *Trouver* $a^* = argmax_{a \in L}(x^*(a))$, *tel que* x^* *est le vecteur propre principal de*

M ;

4. *Si* $x^*(a^*) = 0$ *donner la solution* x*. Si non,* $x(a^*) = 1$ *et retirer* a^* *de* L *;*
5. *Supprimer toutes les affectations qui sont en conflit avec* a^* *(correspondance un-à-un) ;*
6. *Si* L *est vide donner la solution* x*. Si non, refaire l'étape 3 et 4 ;*

4.3.5 Résultats et discussion

Les résultats présentés dans cette partie, sont obtenus à partir des expériences réalisées sur la même base de données utilisée, dans la section précédente. La comparaison des résultats donnés par le tableau 4.3, met en évidence l'amélioration du taux de reconnaissance grâce aux modifications apportées au premier système « système 1 » de reconnaissance. En effet, en combinant le détecteur Harris des points clés et la méthode d'appariement inexact des graphes, la majorité des problèmes de confusion sont corrigés comme il est illustré par le tableau 4.4. Seul le problème de confusion du caractère « ya » et « yar » est persistant. En effet, même avec l'œil humain, il est difficile de différencier ces caractères tels qu'ils sont stockés dans la base de données. La figure 4.18 illustre ce fait.

L'erreur de la non-classification vient du fait que la méthode de Harris est sensible aux formes incurvées et au changement de taille ; ce qui en résulte une détection de points clés bien plus que ce qu'est désiré et aussi une augmentation en temps d'exécution.

Tableau 4-3 Taux de reconnaissance, Taux d'erreur et temps d'exécution obtenus

	Système 1	**Système 2**
Taux de reconnaissance (%)	91	99
Taux d'erreur (%)	9	1
Temps d'exécution (sec)	400	3550

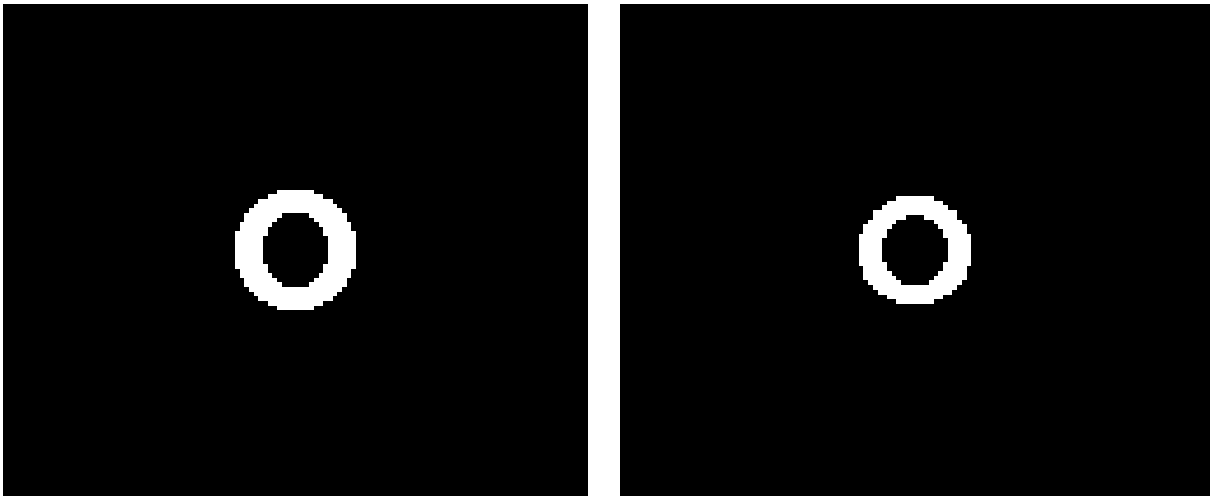


Figure 4-19. Image du caractère ya et yar dans la base de donnée

Tableau 4-4 Taux de reconnaissance, confusion et non-classification du système 2

Nom du caractère	Taux de reconnaissance	Taux de confusion	Taux de non-classification
Ya	91	9	0
Yab	100	0	0
Yag	100	0	0
Yag	100	0	0
Yag^{w}	91	0	9
Yad	100	0	0
Yaḍ	100	0	0
Yey	100	0	0
Yef	100	0	0
Yak	100	0	0
Yak^{w}	97	0	3
Yah	100	0	0
Yaḥ	100	0	0
Yaɛ̣	100	0	0
Yax	100	0	0
Yaq	100	0	0
Yi	98	0	2
Yaj	100	0	0
Yal	100	0	0
Yam	98	0	2
Yan	100	0	0
Yu	100	0	0
Yar	100	0	0
Yaṛ	97	0	3
Yaɣ̄	100	0	0
Yas	100	0	0
Yaṣ	99	0	1
Yac	100	0	0
Yat	100	0	0
Yaṭ	100	0	0
Yaw	100	0	0
Yay	100	0	0
Yaz	97	0	1
Yaẓ	100	0	2

4.3.6 Évaluation

Pour mettre en valeur notre système, nous l'avons comparé avec d'autres systèmes réalisés dans notre laboratoire. Le tableau 4.5 ci-dessous, illustre cette comparaison qui met en évidence les performances de notre système en termes de taux de reconnaissance et de rapidité.

Tableau 4-5 Comparaison des systèmes de reconnaissance des caractères tifinagh imprimés

Auteurs		**Système élaboré**	**Oujaoura et al. [67]**	**Bencharef et al. [66]**	**El Ayachi et al. [63]**
Système de reconnaissance	**Descripteur**	**Méthode de Harris**	**Descripteur Gist**	**descripteurs géodésiques**	**transformée de Walsh**
	Classificateur	**Appariement spectral des graphes**	**Réseaux bayésiens**	**Réseaux de neurone et Arbre de Décision**	**Réseaux de neurones**
Taux de reconnaissance (%)		99	98	93	93
Taux d'erreur (%)		1	2	7	7
Temps d'exécution (sec)		3 550	5 913	-	-

4.3.7 Récapitulatif

Dans cette section, nous avons présenté un système rapide et précis pour la reconnaissance des caractères Tifinagh. Il est basé sur le détecteur de coins Harris et la mise en correspondance des graphes via une méthode d'appariement spectral efficace. Cette méthode utilise les propriétés spectrales de la matrice d'affinité pour calculer le score de similarité entre les graphes. L'utilisation de la matrice d'adjacence nous a permis d'ajouter des poids sur les segments pour une meilleure discrimination. Les résultats des expériences réalisés montrent que la plupart des caractères sont bien reconnus.

4.4 Conclusion

Dans ce chapitre, nous avons présenté nos contributions concernant la reconnaissance hors ligne des caractères Tifinaghs imprimés. Nous avons pu réaliser un système de reconnaissance rapide et précis. En effet, dans notre première tentative, nous avons adopté une approche exacte pour avoir un système rapide. Cependant, celle-ci nécessite plusieurs contraintes pour être applicable en particulier sur les caractères manuscrits. Dans notre deuxième tentative, nous avons adopté une approche inexacte en utilisant une méthode d'appariement spectrale des graphes. Cette approche est renforcée par une amélioration des méthodes de prétraitement et une meilleure représentation graphique en utilisant la méthode de Harris et les matrice d'adjacence qui supportent l'ajout des informations sur les segments. Les expériences réalisées sur une base de caractères imprimés et la comparaison de résultats obtenus avec ceux d'autres travaux, ont mis en évidence les performances prometteuses du système que nous avons élaboré, en termes de taux de reconnaissance et du temps d'exécution.

Les résultats ainsi obtenus, nous ont encouragé à étudier, par la suite, la reconnaissance des caractères Tifinaghs manuscrits.

5 Reconnaissance hors ligne des caractères Tifinaghs manuscrits

5.1 Introduction

Les styles de l'écriture des caractères manuscrits sont très diversifiés. Ils sont souvent ligaturés et leurs graphismes sont inégalement proportionnés. La reconnaissance de ces caractères requiert ainsi des traitements plus discriminatifs que ceux utilisés dans la reconnaissance des caractères imprimés au niveau du prétraitement, de l'extraction des primitives et de la classification. Les travaux de recherche réalisés dans ce contexte, sont généralement dédiés à la reconnaissance des caractères manuscrits Arabes, Latins et Chinois. Quant à la reconnaissance des caractères Tifinaghs, la recherche menée dans ce contexte demeure timide. En effet, selon notre connaissance, depuis l'intégration de cet alphabet dans le système de codage universel en 2004, juste quatre travaux de recherche [118-121] relatifs à ce sujet, sont publiés. Afin de renforcer et d'appuyer ces activités de recherche, nous nous sommes proposés de mener un travail de recherche sur la reconnaissance des caractères Tifinaghs manuscrits par les graphes.

Dans ce travail, nous avons étudié dans un premier temps, les performances de reconnaissance des caractères tifinaghs manuscrits par le système 2. Ensuite, en se basant sur l'analyse des résultats obtenus par cette étude, nous avons élaboré un troisième système « système 3 » plus performant.

5.2 Reconnaissance des caractères Tifinaghs manuscrits par le système 2

Dans cette partie, nous avons utilisé le système 2, décrit dans le chapitre 3, pour la reconnaissance des caractères Tifinaghs manuscrits. Les résultats présentés dans les tableaux 4.1 et 4.2 sont obtenus en utilisant une base de données [122] contenant 25740 images des caractères Tifinaghs manuscrits, dont chaqu'un des alphabets est représenté par 780 images. Cette base représente le style d'écriture de 60 personnes. Durant nos tests, nous avons utilisée 80% de cette base comme images de références.

La comparaison de ces résultats par rapport à ceux obtenus dans le cas de la reconnaissance des caractères imprimés, montre qu'il y a une réduction des taux de reconnaissance et une élévation du temps d'exécution. Le système 2 n'est donc pas un système performant pour la reconnaissance des caractères Tifinaghs manuscrits. En effet, l'algorithme d'extraction des

points clés détecte un nombre de points plus grand que le nombre optimal de points nécessaire pour la représentation graphique des caractères manuscrits. Cela affecte la discrimination entre ces caractères par la méthode d'appariement inexact adoptée, bien qu'elle tolère la dissimilarité. La recherche d'une solution à cette problématique, par la suite, a abouti à l'élaboration d'un nouveau système « système 3 » pour la reconnaissance des caractères Tifinagh manuscrits.

Tableau 5-1 Résultats obtenus par le système 2 sur les caractères imprimés et manuscrits

	Caractères manuscrits	**Caractères imprimés**
Taux de reconnaissance (%)	85	99
Taux d'erreur (%)	15	1
Temps d'exécution (sec)	25563	3550

Tableau 5-2 Taux de reconnaissance, confusion et non-classification du système 2, cas du manuscrit

Nom du caractère	**Taux de reconnaissance**	**Taux de confusion**	**Taux de non-classification**
Ya	80	11	9
Yab	90	8	2
Yag	97	0	3
Yag^{w}	91	0	9
Yad	99	1	0
Yaḍ	97	2	1
Yey	91	4	5
Yef	79	21	0
Yak	65	22	13
Yak^{w}	97	0	3
Yah	80	20	0
Yaḥ	91	9	0
Yaɛ̣	93	7	0
Yax	75	20	5
Yaq	70	29	1
Yi	98	0	2
Yaj	83	17	0
Yal	72	20	8
Yam	98	0	2
Yan	85	15	0
Yu	76	8	16
Yar	81	11	8
Yaṛ	97	0	3
Yaɣ	78	11	11
Yas	88	12	0
Yaṣ	99	0	1
Yac	70	24	6
Yat	82	16	2
Yaṭ	87	10	3
Yaw	85	10	5
Yay	72	19	9
Yaz	89	5	6
Yaẓ	79	2	19

5.3 Élaboration d'un nouveau système pour la reconnaissance des caractères Tifinaghs manuscrits

5.3.1 Description du système

Dans cette partie, nous décrivons le système de reconnaissance des caractères manuscrits que nous avons développé en proposant une nouvelle approche pour la détection des points clés et pour la classification. Il est composé de trois phases principales : prétraitement, représentation structurelle et appariement des graphes. La figure 5.1, illustre les fonctions utilisées au niveau de chaque phase.

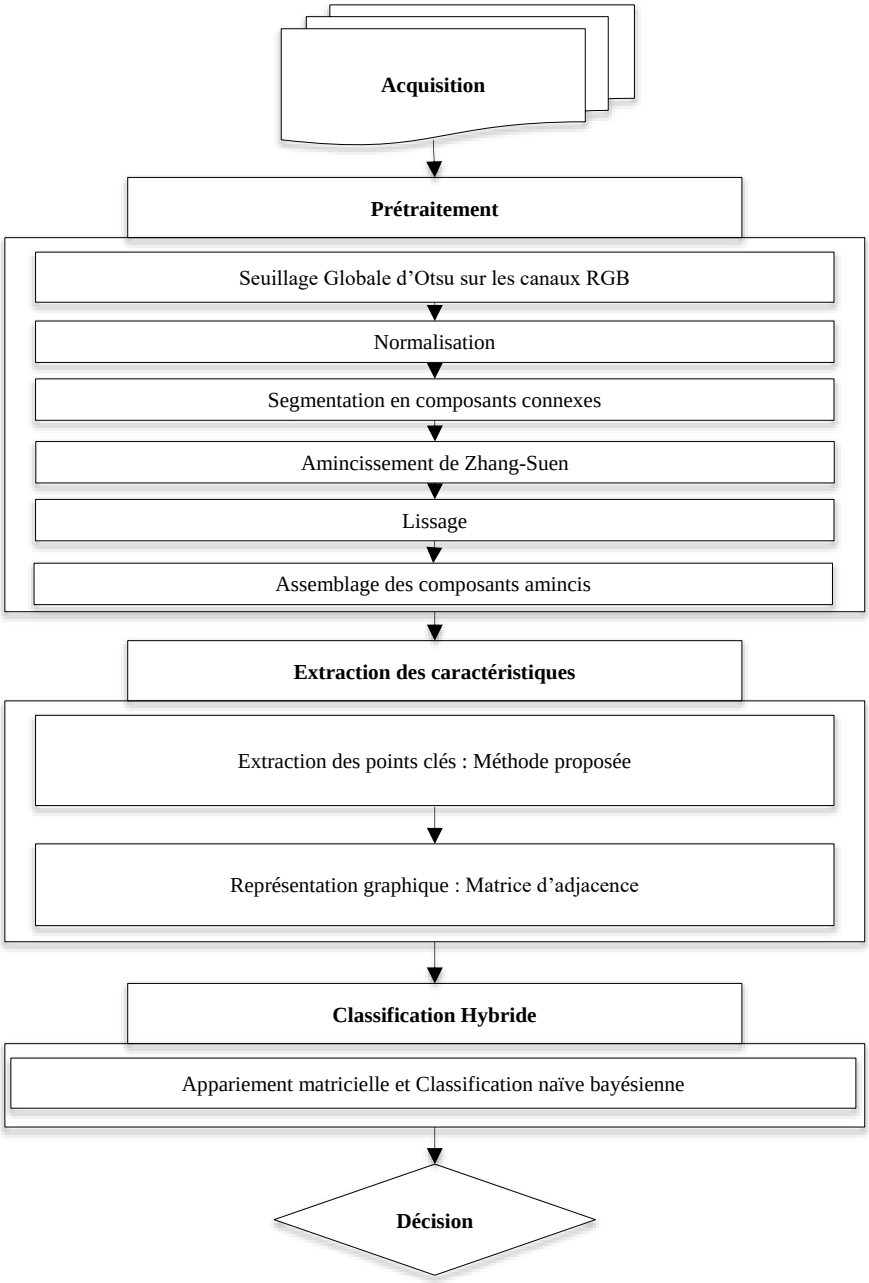


Figure 5-1. Système de reconnaissance des caractères Tifinaghs élaboré

5.3.1.1 Prétraitements

Le processus que nous avons adopté pour le prétraitement des caractères Tifinaghs manuscrits, est illustré par la figure 5.2. Dans cette phase, nous avons commencé par séparer les plans rouge, vert et bleu à partir de l'image couleur d'entrée. Le résultat de cette séparation est un ensemble de trois images en niveau de gris. Ensuite, nous avons appliqué l'algorithme de seuillage globale d'Otsu à ces trois images pour les transformées en images binaires. Ces image sont fusionnées à l'aide d'un opérateur OU logique, pour fournir une image binaire finale. Cette image est filtrée en éliminant les structures légères liées aux bords de l'image, c'est-à-dire les structures qui sont 8-connectés à la bordure de l'image, et les pixels isolés dont le nombre de pixel d'avant plan, dans les 8-voisinages, est nuls éliminés. Ce filtrage est illustré dans la figure 5.3. Enfin, une normalisation et un amincissement sont appliqués comme il est décrit dans la section 4.3.1 dans le chapitre précédant.

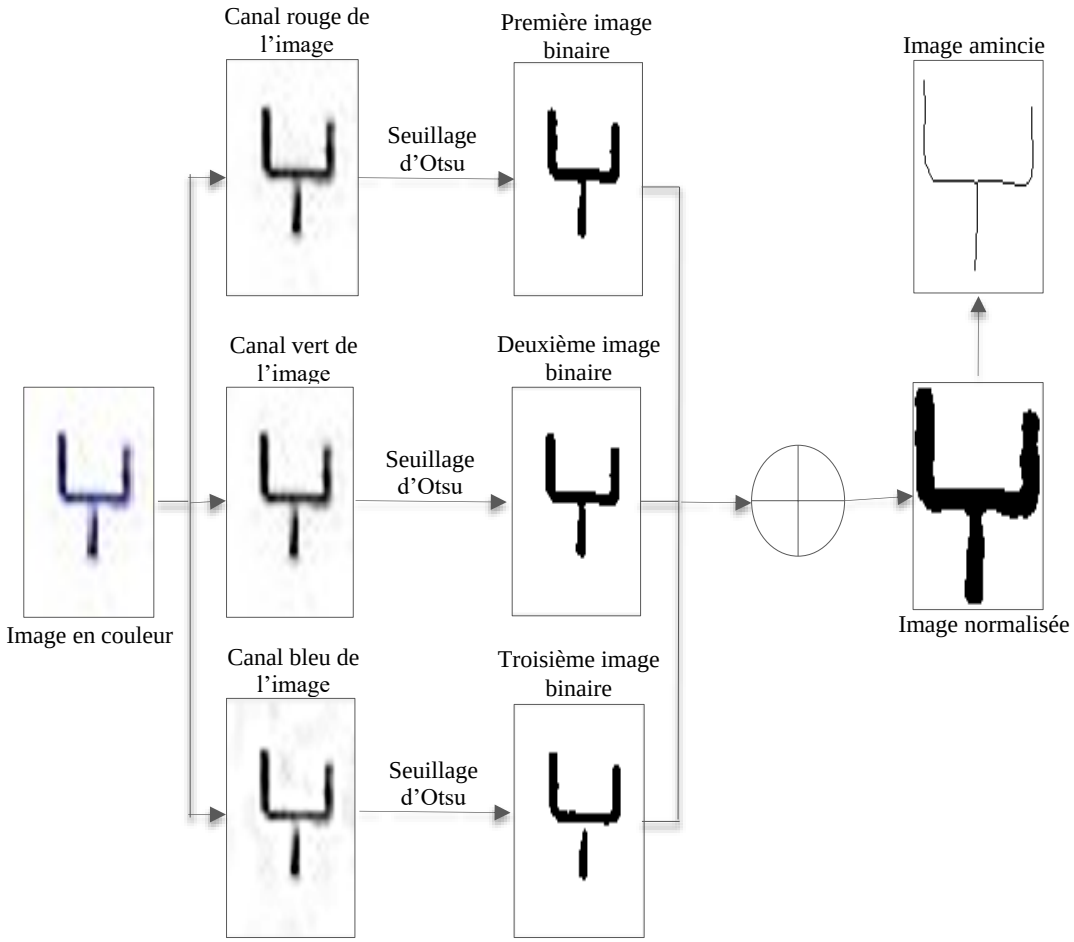
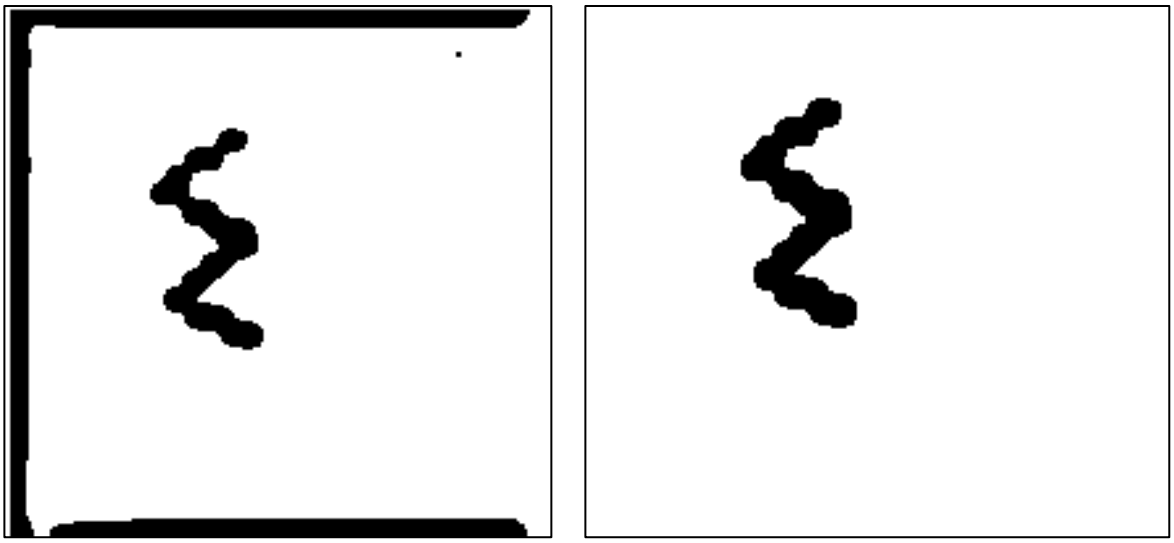


Figure 5-2 Processus de prétraitements des caractères Tifinaghs manuscrits, exemple du caractère yaɣ



(a) (b)

Figure 5-3 Élimination du bruit de frontière et les pixels isolés, (a) image binaire, (b) image binaire filtré

La figure 5.4 ci-dessous, illustre le temps d'exécution du processus de prétraitements des caractères Tifinaghs manuscrits et ceux des caractères imprimés. Le temps de prétraitements du manuscrit est évidemment plus grand que celui de l'imprimé puisque ce dernier nécessite beaucoup plus de fonctions de prétraitements.

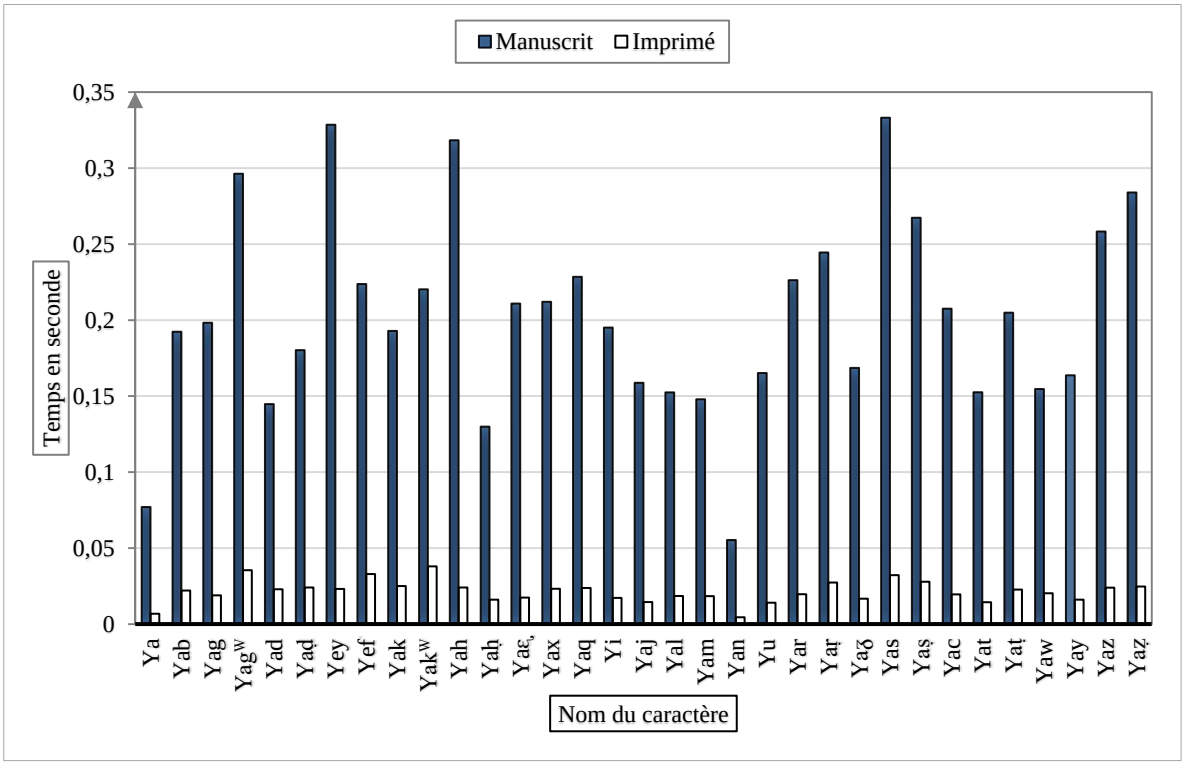


Figure 5-4 Temps de prétraitement des caractères manuscrits et imprimés par le processus adopté

5.3.1.2 Représentation structurelle

5.3.1.2.1 Nouvelle approche de détection des points clés

La détection des points clés par la méthode de Harris est sensible face aux formes incurvées, surtout dans le cas du manuscrit ou l'écriture est rarement droite. Généralement le nombre de points clés détecté dépasse le nombre de points optimal nécessaire pour la représentation graphique des caractères. En effet, lorsque les relations entre les points clés, les positions des segments, leurs longueurs et leurs orientations sont prisent en compte, la graphie de l'alphabet Tifinaghs peut être représenté par un nombre de points optimal en restant discriminant, comme il est présenté dans la figure 5.5 ci-dessous.

Graphème	Représentation graphique	Nombre de points - segments	Graphème	Représentation graphique	Nombre de points - segments	Graphème	Représentation graphique	Nombre de points - segments
ⵔ		4 - 4	ⵅ		4 - 3	ⵇ		6 - 6
ⵙ		4 - 5	ⵛ		6 - 5	ⵜ		6 - 5
ⵠ		6 - 6	ⵡ		7 - 6	ⵣ		6 - 5
ⵠⵯ		10 - 9	ⵆ		4 - 4	ⵈ		6 - 6
ⵏ		3 - 2	ⵚ		5 - 4	ⵄ		9 - 10
ⵉ		6 - 5	ⵊ		6 - 5	ⵟ		5 - 4
ⵒ		10 - 9	ⵍ		4 - 3	ⵋ		9 - 8
ⵀ		10 - 8	ⵔ		4 - 3	ⵓ		4 - 3
ⵅ		5 - 5	ⵏ		2 - 1	ⵙ		4 - 3
ⵅⵯ		9 - 8	ⵒ		8 - 8	ⵣ		8 - 7
ⵐ		4 - 5	ⵔ		4 - 4	ⵣ		11- 10

Figure 5-5 Exemple de nombre de points et segments pour chaque graphème de l'alphabet Amazighe

Afin d'avoir la représentation graphique souhaitable des caractères, nous avons proposé un algorithme de segmentation des caractères permettant de remédier aux limites du détecteur des coins Harris. Cet algorithme consiste à détecter, pour chaque caractère, des points clés qui se subdivisent en deux classes : points clés primaires et points clés secondaires.

Dans un premier temps, les points clés primaires sont extraits. Il s'agit des points d'intersection (nombre de transitions est supérieur à 2) et des points d'extrémités ou de fin (nombre de transitions est égal à 1). La section 4.2.2.1, décrit l'algorithme d'extraction de ce type de points. La figure 5.6 ci-dessous, présente les points clés primaires extraits du caractère yaɣ.

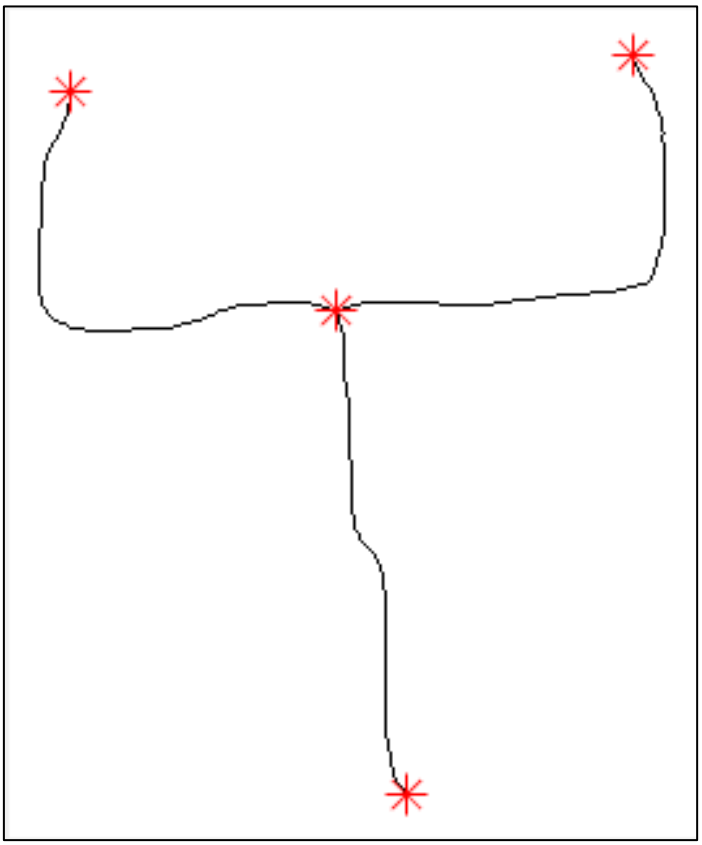


Figure 5-6 points clés primaires du caractère yaɣ

Dans un deuxième temps, le caractère est divisé en plusieurs segments en utilisant les points clé primaires comme délimiteurs (voir figure 5.7 ci-dessous). Cela, nous permet d'analyser chaque segment séparément et décider si un point clé secondaire est nécessaire ou non.

Nous avons proposé un critère qui permet de classer les segments en deux types de formes géométriques simples : lignes droites et arc. Il s'agit de calculer la longueur du segment D par

rapport à la distance euclidienne entre les extrémités de celui-ci. Si ce rapport est supérieur à un seuil k, alors le segment est un arc ; si non, le segment est une droite. Ce seuil nous permet non seulement de déterminer la nécessité d'ajout d'un point secondaire dans un segment, mais aussi de contrôler le nombre de points secondaire qu'on veut ajouter. Dans notre cas, cette valeur est, choisie empiriquement, égale à 0,2.

$$\begin{cases} \frac{D-d}{d} \geq k \ le\ segment\ est\ un\ arc \\ \frac{D-d}{d} < k \ le\ segment\ est\ une\ ligne\ droite \end{cases} \quad (5.1)$$

Lorsqu'un arc est détecté, la distance euclidienne orthogonale entre les éléments du segment arc et la ligne droite connectant les extrémités de celui-ci est calculée. L'élément qui possède la distance maximale est désigné comme un point secondaire. Chaque fois qu'un point est ajouté dans un segment, deux nouveaux segments remplacent l'ancien segment dans la liste des segments. La figure 5.8 ci-dessous, illustre les points clés secondaires insérés dans la liste des points clés.

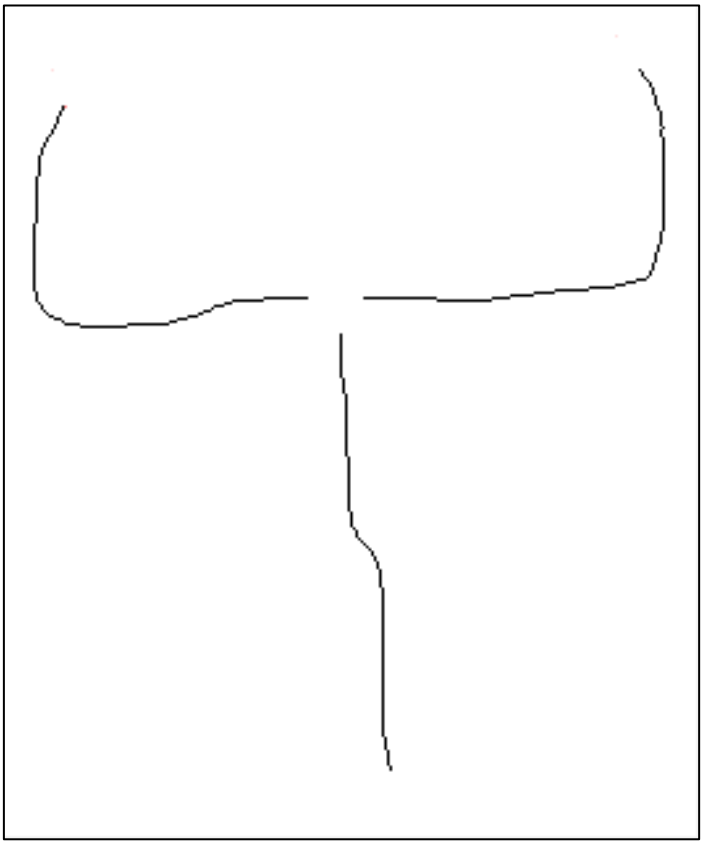


Figure 5-7 La première segmentation du caractère yay

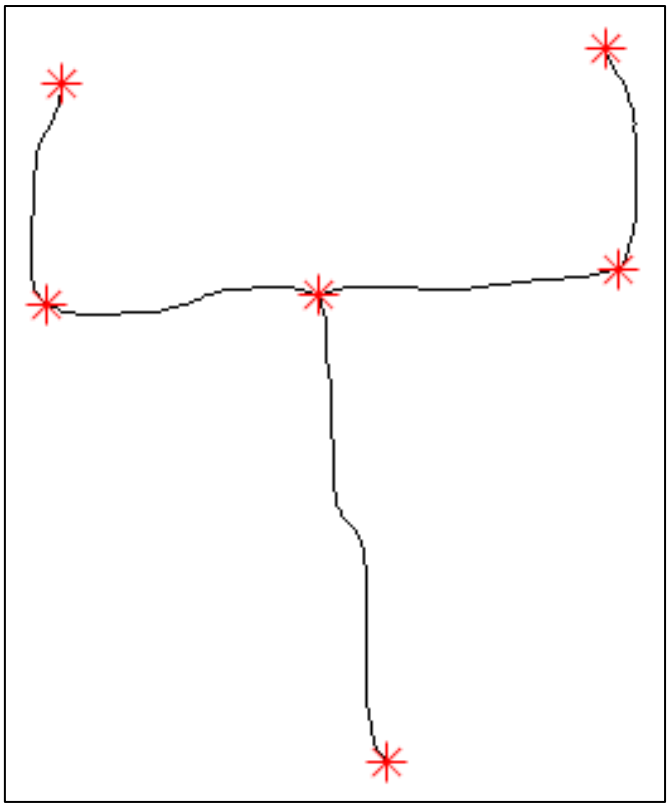


Figure 5-8 Points clés finaux du caractère yaɣ

L'algorithme utilisé pour l'extraction des points est le suivant :

1. *Extraire les points d'intersections et points d'extrémité ;*
2. *Extraire les segments reliant ces points ;*
3. *Pour chaque segment, vérifier si c'est un arc ou ligne droite ;*
4. *Pour chaque arc designer le pixel qui a la plus grande distance orthogonale comme un point singulier ;*
5. *Mettre à jour la liste des segments ;*
6. *Tant qu'il y a un segment de type arc répéter l'étape 3 à 5.*

Afin d'évaluer les performances de l'algorithme ainsi proposé, nous avons comparé, pour chaque caractère Tifinagh manuscrit, d'une part, le nombre de points clés extraits par cet algorithme à celui extrait par l'algorithme de Harris et au nombre optimal de points (donné par la figure 5.5) nécessaires pour la représentation graphique ; et d'autre part, les temps

d'exécution de ces algorithmes. Cette comparaison, illustrée par les figures 5.9 et 5.10, montre bien que notre algorithme est plus performant que celui de Harris aussi bien au niveau de la rapidité qu'au niveau de l'extraction des points clés juste nécessaires pour la représentation des caractères Tifinaghs manuscrits.

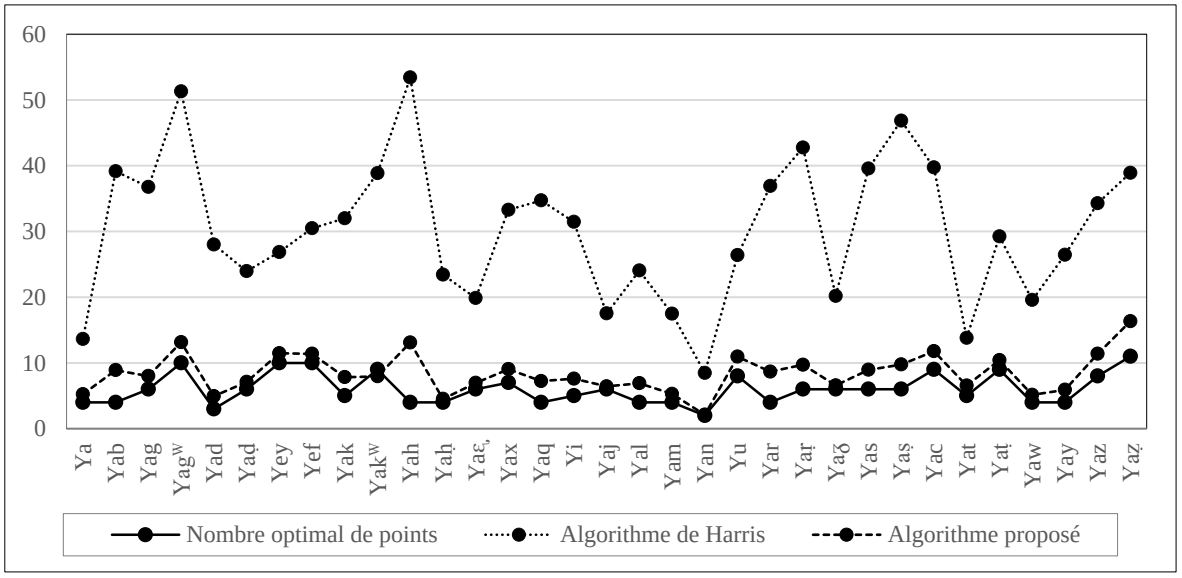


Figure 5-9 Comparaison des nombre de points obtenus par la méthode de Harris, l'algorithme proposé et le nombre de points nécessaires

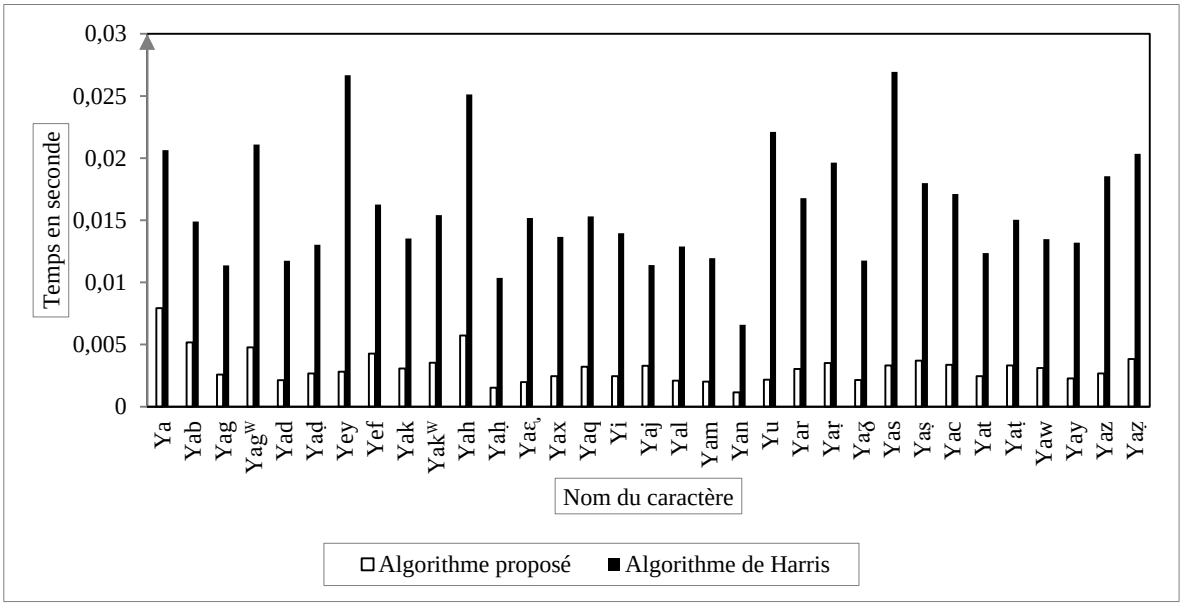


Figure 5-10 Temps d'exécution de l'algorithme de Harris et de l'algorithme proposé, cas du manuscrit

5.3.1.2.2 Représentation graphique

Les points clés extraits par l'algorithme proposé, sont utilisés pour la représentation graphique des caractères à l'aide de l'algorithme de construction graphique décrit dans la section 4.3.3. La figure 5.11 ci-dessous, illustre la représentation graphique du caractère yaɣ en se basant sur les coordonnées des points clés dans l'image originale. Cette représentation est plus rapide que dans le cas de l'utilisation des points clés extraits par Harris. Ceci est illustré par la figure 5.12 qui donne le temps d'exécution de l'algorithme de la représentation graphique pour chaque caractère.

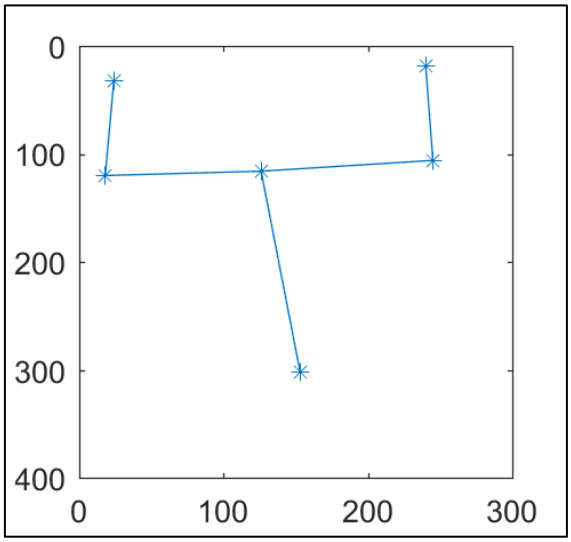


Figure 5-11. Représentation graphique du caractère yaɣ

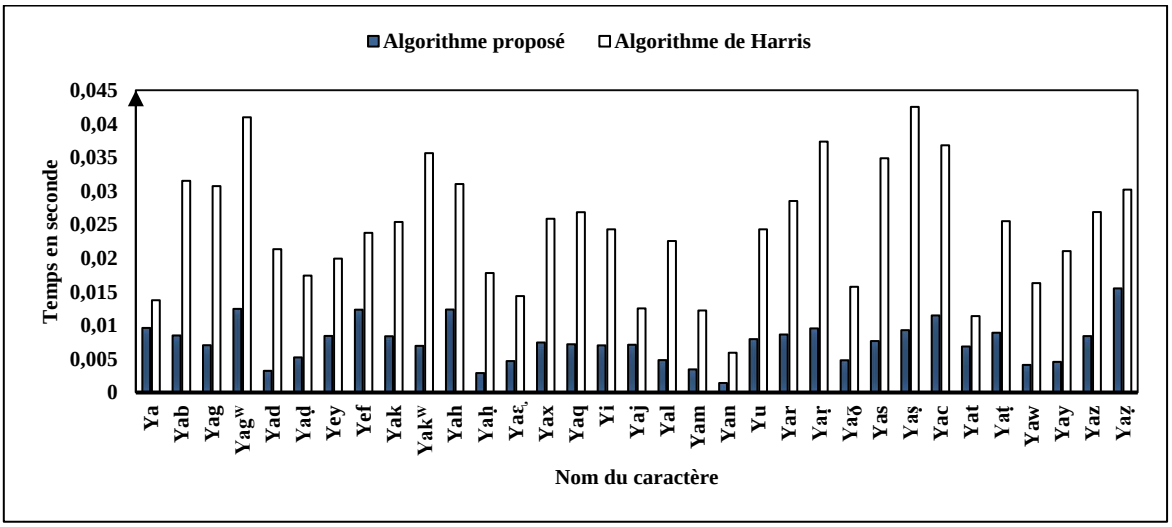


Figure 5-12. Temps d'exécution de l'algorithme de la représentation graphique des caractères par les points clés extraits par l'algorithme proposé et l'algorithme de Harris

5.3.1.3 Classification hybride

Dans cette phase, la représentation graphique obtenue d'une image d'entrée est comparée avec celle des images de références. Afin de bénéficier de la puissance représentationnelle des graphes et de la robustesse des méthodes statistiques de classification sans perdre en temps d'exécution, nous proposons une classification hybride qui se base sur la comparaison matricielle et la classification naïve bayésienne. Cette hybridation est effectuée de façon séquentielle comme illustré dans la figure 5.13 ci-dessous. La classification naïve bayésienne est utilisée comme alternative au cas où la comparaison matricielle donne plusieurs classes pour affiner d'avantage la classification.

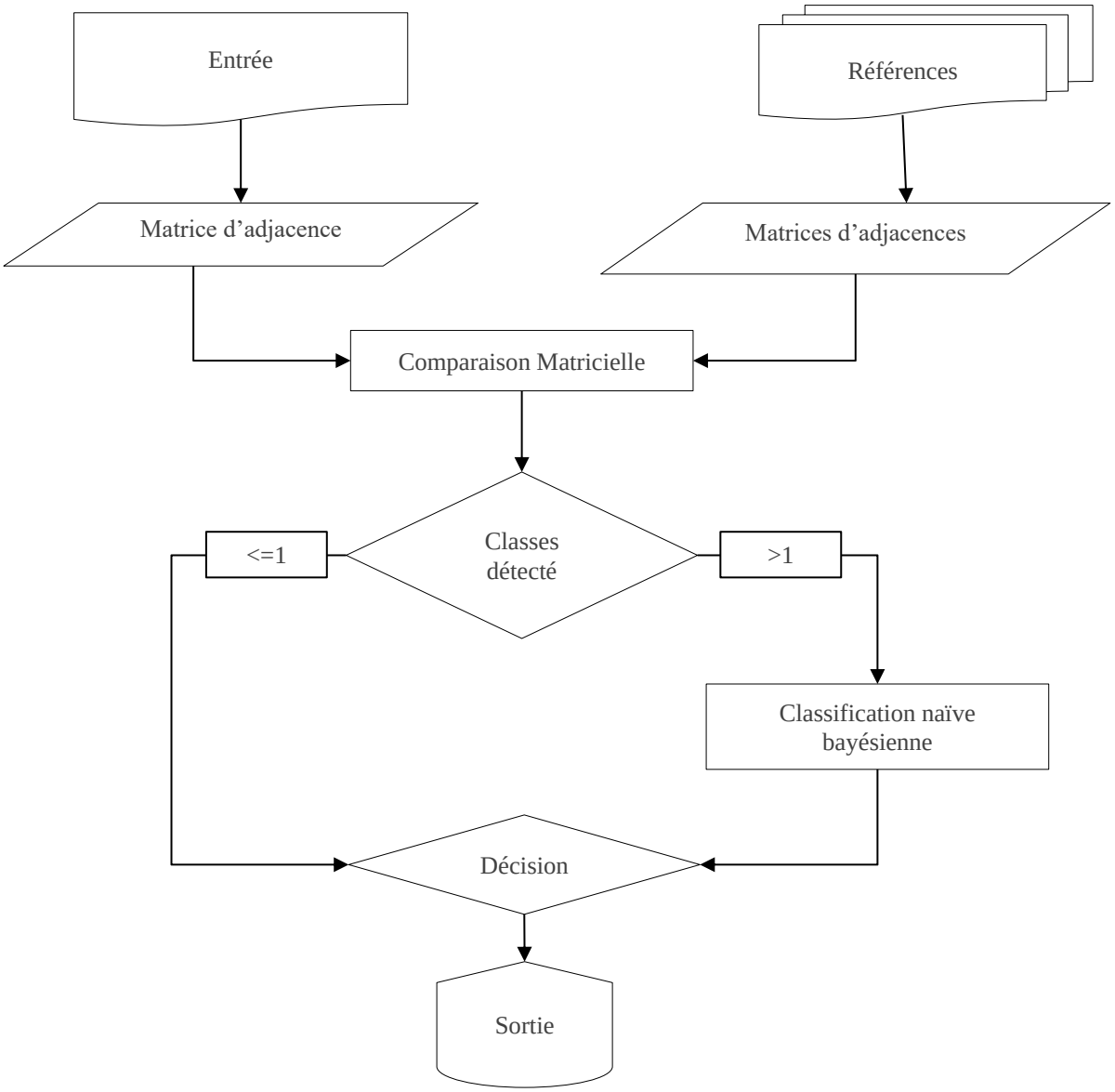


Figure 5-13 Processus de classification hybride proposé

5.3.1.3.1 Caractéristique de la représentation matricielle

La représentation matricielle par des matrices d'adjacence nous permet, non seulement de représenter la structure du caractère, mais aussi d'ajouter des poids sur les segments qui définies cette représentation ; ce qui permet d'éviter de refaire le calcul des paramètres des segments composants les caractères et de réduire par conséquence la durée des traitements au cours de la classification. D'autre part, les valeurs non nulles de ces matrices permettent la construction des vecteurs caractéristiques fournis au classificateur bayésien lorsque la comparaison matricielle trouve plusieurs classes de l'image d'entrée.

5.3.1.4 Classification naïve bayésienne

Le classificateur bayésien naïf [123] est un schéma de classification simple, qui estime la probabilité conditionnelle de classe en supposant que les attributs sont conditionnellement indépendants, étant donné l'étiquette de classe c. L'hypothèse d'indépendance conditionnelle peut être formellement énoncée comme suit:

$$P(A/C = c) = \prod_{i=1}^{n} P(A_i/C = c) \qquad (5.2)$$

Chaque ensemble d'attributs A est constitué de n valeurs d'attributs :

$$A = \{A_1, A_2, \dots, A_n\} \qquad (5.3)$$

Avec l'hypothèse d'indépendance conditionnelle, l'algorithme estime seulement la probabilité conditionnelle de chaque A_i au lieu de calculer la probabilité conditionnelle de classe pour chaque groupe d'attributs A. Cette dernière approche est plus pratique car elle ne nécessite pas un ensemble d'entraînement très important pour obtenir une bonne estimation de la probabilité. Pour classer un exemple de test, le classificateur bayésien naïf calcule la probabilité à posteriori pour chaque classe C.

$$P(C/A) = \frac{P(C)\prod_{i=1}^{n} P(A_i/C)}{P(A)} \qquad (5.4)$$

Puisque $P(A)$ est fixe pour tout A, il suffit de choisir la classe qui maximise le terme du numérateur : $P(C)\prod_{i=1}^{n} P(A_i/C)$.

Le classificateur bayésien naïf peut :

- Prendre en compte des donnée manquantes ;
- Ajouter facilement de nouvelles classes, il suffit d'en apprendre le modèle ;
- Prendre en compte la composition avec le minimum d'exemples possibles durant l'entrainement.

5.3.2 Résultats et discussion

Les résultats concernant la reconnaissance des caractères Tifinaghs manuscrits par le nouveau système réalisé sont illustrés par les tableaux 5.3 et 5.4. Ces résultats obtenus par les expériences réalisées en utilisant la base de données AMHCD, décrite dans la section 5.2.

La comparaison de ces résultats avec ceux obtenus par le système 2, met en évidence la nette amélioration du temps d'exécution et du taux reconnaissance. Le taux d'erreur obtenus est dû essentiellement à la non-classification car le taux de confusion est très faible.

Tableau 5-3 Taux de reconnaissance des caractères manuscrits

Système de reconnaissance	**Système 3**	**Système 2**
Algorithme d'extraction des points clés	**Algorithme proposé**	**Harris**
Descripteur	**Matrice d'adjacence**	**Matrice d'adjacence**
Classificateur	**Appariement matricielle exact et classification bayésienne naïve**	**Appariement spectral des graphes**
Taux de reconnaissance (%)	96	85
Taux d'erreur (%)	4	15
Temps d'exécution (sec)	2000	25563

Tableau 5-4 Taux de reconnaissance, confusion et de non classification de chaqu'un des caractères Tifinaghs manuscrits, obtenus par le système 3

Taux (%) / Nom du caractère	Reconnaissance	Confusion	Non-classification
Ya	99	1	0
Yab	100	0	0
Yag	94	0	6
Yag^{w}	97	0	3
Yad	100	0	0
Yaḍ	100	0	0
Yey	99	1	0
Yef	100	0	0
Yak	90	0	10
Yak^{w}	95	0	5
Yah	100	0	0
Yaḥ	100	0	0
Yaɛ̣	100	0	0
Yax	100	0	0
Yaq	96	0	4
Yi	94	2	4
Yaj	100	0	0
Yal	100	0	0
Yam	97	0	3
Yan	95	0	5
Yu	100	0	0
Yar	98	2	0
Yaṛ	97	0	3
Yaǧ	100	0	0
Yas	90	0	10
Yaṣ	98	1	1
Yac	85	0	15
Yat	100	0	0
Yaṭ	90	0	10
Yaw	100	0	0
Yay	95	1	4
Yaz	80	0	20
Yaẓ	85	0	15

5.3.3 Évaluation

Pour mettre en valeur notre système, nous l'avons comparé avec d'autres systèmes réalisés au sein de notre laboratoire. Le tableau 5.5 ci-dessous, illustre cette comparaison en termes de taux de reconnaissance (rappel) et temps d'exécution. D'après ce tableau comparatif, il apparait clairement que notre système le plus performant.

Tableau 5-5 Comparaison des systèmes de reconnaissance des caractères tifinagh manuscrits

Auteurs		**Système élaboré**	**El Kessab et al. [70]**	**Gounane et al [71]**
Système de reconnaissance	**Descripteur**	**Algorithme proposée et Matrice d'adjacences**	**Morphologie Mathématique**	**(zonage) Distance du centre de gravité**
	Classifieur	**Classification hybride proposée**	**Perceptrons multicouches et HMM**	**K-NN et B-Gram**
Taux de reconnaissance (%)		96	92	91
Taux d'erreur (%)		4	8	9
Temps d'exécution (en seconde)		2000	-	-

5.4 Conclusion

Dans ce chapitre, nous avons présenté notre contribution concernant la reconnaissance hors ligne des caractères Tifinaghs manuscrits. Il s'agit de la réalisation d'un système de reconnaissance de ces caractères par les graphes. Lors de l'élaboration de ce système, nous avons dans un premier temps, développé une méthode plus performante que celle de Harris, pour l'extraction des points clés nécessaires à la représentation graphique des caractères. Dans un deuxième temps, nous avons proposé une méthode de classification hybride combinant l'appariement exact et réseau bayésien naïf. La mise en œuvre du système, intégrant ces deux méthodes, nous a permis d'évaluer ses performances par rapport à ceux des systèmes existants. Cette évaluation a mis en évidence que le système, que nous avons réalisé, est le plus performant en termes de taux de reconnaissance et du temps d'exécution.

6 Conclusion générale et perspectives

6.1 Conclusion générale

Dans ce manuscrit, nous avons présenté le travail de recherche que nous avons mené sur la reconnaissance des caractères Tifinaghs. Il s'agit de l'élaboration d'outils de prétraitement, de description par les graphes et de classification permettant la réalisation des système de reconnaissance hors ligne des caractères imprimés et manuscrits.

En ce qui concerne les caractères imprimés, nous avons dans un premier temps, élaboré un système à base des outils de traitement permettant :

- La représentation des caractères par les graphes à l'aide des points clés extraits par la méthode des voisinages à partir de leurs squelettes obtenus par l'algorithme d'amincissement Zhang-Suen ;
- La classification par l'appariement exact des graphes en comparant les matrices d'incidence.

La mise en œuvre de ce système de reconnaissance à l'aide d'une base de données, a montré que son temps d'exécution est très satisfaisant, par contre son taux de reconnaissance (91%) n'est pas à la hauteur des attentes. Afin d'améliorer ce taux, nous avons, dans un deuxième temps, adopté les outils réalisant les traitements suivant :

- Représentation graphique des caractères par les points clés détectés par la méthode de Harris à partir de leurs squelettes soigneusement lissé via une procédure que nous avons élaborée ;
- Description des graphes par des matrices d'adjacence ;
- Classification par l'appariement inexact des graphes à l'aide des spectres de leurs matrices d'adjacence.

La mise en œuvre du nouveau système intégrant ces outils, a montré la nette amélioration du taux de reconnaissance (99%). Cependant, lorsque nous avons utilisé ce système pour la reconnaissance des caractères Tifinaghs manuscrits, le temps d'exécution a augmenté et le taux de reconnaissance a diminué (85%). Pour résoudre ce problème concernant la reconnaissance des caractères manuscrits, il a fallu chercher à développer des méthodes de

traitement permettant une description rapide et précis, et une classification plus discriminante. Le travail réalisé dans ce sens, nous a permis d'élaborer un système de reconnaissance incluant des méthodes de traitement plus avancées qui consistent à :

- Extraire, par une nouvelle méthode plus efficace que celle de Harris, le nombre optimal des points clés suffisants pour la construction du graphe d'un caractère ;
- Classifier les caractères par combinaison de l'appariement exact des graphes avec le réseau bayésien naïf.

L'amélioration de l'efficacité de la reconnaissance des caractères Tifinaghs manuscrits par ces méthodes, est mise en évidence par des expériences réalisées sur la base de données des caractères amazighes manuscrits AMHCD.

L'analyse des résultats des travaux antérieurs, réalisés sur la reconnaissance des caractères Tifinaghs, a fait apparaitre que notre système est le plus performant en termes du temps d'exécution et du taux de reconnaissance. .

6.2 Perspectives

Le travail que nous avons présenté dans ce manuscrit peut être poursuivi selon trois orientations :

- **Amélioration des systèmes de reconnaissances développés**

  Bien que les résultats obtenus au cours de ce travail sont prometteurs, Il existe plusieurs axes d'amélioration du processus de reconnaissance des caractères Tifinaghs par les graphes, qui consistent en :

  - L'intégration des techniques pour récupérer les informations perdues l'or de l'acquisition des images des caractères ;
  - L'automatisation du calcul du seuil de l'algorithme proposé pour l'extraction des points clés ;
  - L'utilisation d'autres approches pour la représentation et l'appariement des graphes.
- **Reconnaissance des documents textes Tifinaghs**

- **Application des graphes en traitement d'images**

Références

[1] M. Ameur, A. Bouhjar, F. Boukhris, A. Boukouss, A. Boumalk, M. Elmedlaoui, E. Iazzi, H. Souifi, "Initiation à la langue Amazighe", Publications de l'IRCAM, CAL, Rabat, 2004.

[2] Al Falou Wassim, « Reconnaissance de caractères manuscrits par réseau de neurones », Thèse doctorat en informatique, sous la direction de El-Eter Bassam, Ecole Polytechnique Fédérale de Lausanne, 1998, Pages 6-7.

[3] P. Sahoo, S. Soltani and A. Wong, "A Survey of Thresholding Techniques", Computer Vision Graphics Image Processing, Vol. 41, pp. 233-260, 1988.

[4] N. Otsu, "A Threshold Selection Method From Grey-level Histograms", IEEE Transactions on systems, Man and Cybernetics, SMC-9, pp. 62-66, 1979.

[5] P. Rosin and E. Ioannidis, "Evaluation of Global Image Thresholding for Change Detection", Pattern Recognition Letters, Vol. 24, pp. 2345-2356, 2003.

[6] O. Trier and A. Jain, "Goal-Directed Evaluation of Binarization Methods", IEEE Trans. On Pattern Recognition and Machine Intelligence, Vol. 17, No. 12, pp. 1191-1201, 1995.

[7] S. Fischer, "Digital Image Processing: Skewing and Thresholding", Master of Science thesis, University of New South Wales, Sydney, Australia, 2000.

[8] V. M. Mohan, R. K. Durga, S. Devathi, and K. S. Raju. "Image Processing Representation Using Binary Image; Grayscale, Color Image, and Histogram". *In Proceedings of the Second International Conference on Computer and Communication Technologies* (pp. 353-361). Springer India, 2016.

[9] J. C. Russ, "The image processing handbook". *CRC press*. 2016.

[10] L. O'Gorman, M. Sammon and M. Seul, "Practical Algorithms for Image Analysis", Cambridge University Press, ISBN 978-0=521-88411-2, New York, NY, USA, 2008.

[11] J. J. Hull and S.L. Taylor, Document Image Skew Detection Survey and Annotated Bibliography, Document Analysis Systems II, Eds., World Scientific, pp. 40-64, 1998.

[12] A. Al-Shatnawi and K. Omar, "Skew Detection and Correction Technique for Arabic Document Images Based on Centre of Gravity", Journal of Computer Science 5 (5), pp. 363-368, ISSN 1549-3636, 2009.

[13] D. Phillips, "Image Processing", in C. Electronic Edition 1.0, 1st Edition was published by R & D Publications, ISBN 0-13-104548-2, Lawrence, Kansas, USA.2000.

[14] M. Sinecen, “Digital Image Processing with MATLAB”. In Applications from Engineering with MATLAB Concepts. InTech. 2016.

[15] E. Davies, “Machine Vision – Theory Algorithms Practicalities, Third Edition”, Morgan Kaufmann Publishers, ISBN 13: 978-0-12-206093-9, ISBN-10: 0-12-206093-8, San Francisco, CA, USA, 2005.

[16] I. S. Oh, J. S. Lee, C. Y. Suen, “Analysis of class separation and Combination of Class-Dependent Features for Handwriting Recognition”, IEEE Trans. Pattern Analysis and Machine Intelligence, vol.21, no.10, pp.1089-1094, 1999.

[17] D. Trier, A. K. Jain, T. Taxt, “Feature Extraction Method for Character Recognition - A Survey”, Pattern recognition, vol.29, no.4, pp.641-662, 1996.

[18] Chipara, C. Lu, Bailey T, “Reliability clinical monitoring using wireless sensor networks: experiences in a step-down hospital unit”, SenSys 2010.

[19] S. Lukman, Y. He, S. Hui, “Computational methods for traditional Chinese medicine: a survey”. Computer Methods and Programs in Biomedicine, pp 283-293, 2007.

[20] D. Impedovo and G. Pirlo, “Zoning methods for handwritten character recognition: A survey”. Pattern Recognition, 47(3), 969-981, 2014.

[21] M. K. Hu, “Visual pattern recognition by moment invariants”, IRE Transactions on Information Theory, Vol. 8(2), pp. 179-187, 1962.

[22] C. H. Teh and R. T. Chin, “On image analysis by the method of moments”, IEEE Transactions on PAMI, Vol. 10(4), pp. 496- 513, 1988.

[23] C. Y. Suen, M. Berthod and S. Mori, “Automatic recognition of hand printed characters-the state of the art”, Proceedings of the IEEE, Vol. 68(4), pp. 469-487, 1980.

[24] M. H. Glauberman, “Character recognition for business machines”, Electronics, Vol.29, pp. 132-136, 1956.

[25] R. Tarling and R. Rohwer, “Efficient use of training data in the n-tuple recognition method”, Electronics Letters, Vol. 29(24), pp. 2093-2094, 1993.

[26] R. El-Hajj, C. Mokbel and L. Likforman, “HMM-based Arabic Cursive Handwritten Recognition System”, The RTST conference (Int’l. Conference on Research Trends in Science and Technology). LAU University Beirut Lebanon, March 2005.

[27] R. Al-Hajj, “Reconnaissance hors ligne de textes manuscrits cursifs par l’utilisation de systèmes hybrides et de techniques d’apprentissage automatique”. Thèse de Doctorat, Ecole Nationale Supérieure de Télécommunications, Paris, 2007.

[28] R. Kapoor, D. Bagai, and T. Kamal, “Representation and extraction of nodal features of DevNagri letters”. ICVGIP, 2003.

[29] U. Pal, A. Belaïd, and C. Choisy, “Water reservoir based approach for touching numeral segmentation”. International Conference on Document Analysis and Recognition, 892, 2001.

[30] Trier, Øivind Due, JAIN, K. Anil, Torfinn. Taxt, “Feature extraction methods for character recognition-a survey”. Pattern recognition, 1996, vol. 29, no 4, p. 641-662.

[31] Y. Li, S. Wang, Q. Tian and X. Ding, “A survey of recent advances in visual feature detection”. Neurocomputing, 149, 736-751, 2015.

[32] Cabrelli CA, Molter UM, “Automatic representation of binary images,” IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 12, No.12 pp.1190-1196, 1990.

[33] X. Peng, C. Zhou, M. Ding, “Corner detection method based on wavelet transform,” In: Proc. SPIE, Vol. 4550, pp.319-323, 2001.

[34] I. N. Bankman, E. W. Rogala, “Corner detection for identification of man-made objects in noisy aerial images,” In: Proc. SPIE, Vol. 4726, pp. 304-309, 2002.

[35] R. Elias, R. Laganiere, “Cones: a new approach towards corner detection,” Proc Canadian Conference on Electrical and Computer Engineering, Vol. 2, pp.912-916, 2002.

[36] S. Bae, I. S. Kweon, C.D. Yoo, “COP: a new corner detector,” Pattern Recognition Letters, Vol. 23, No. 2, pp.1349-1360, 2002.

[37] A. Quddus, M. Gabbouj, “Wavelet-based corner detection technique using optimal scale,” Pattern Recognition Letter, Vol. 23, pp.215-220, 2002.

[38] S. Smith, J. Brady, “SUSAN—A new approach to lowlevel image processing,” International Journal of Computer Vision, Vol. 23, No.1, pp.45-48, 1997.

[39] C. Harris, M. Stephens. “A combined corner and edge detector,” Proceedings of the Fourth Alvey Vision Conference, University of Sheffield Printing Unit, Manchester, pp.147-151, 1988.

[40] U. Pal, T. Wakabayashi, F. Kimura, “Comparative Study of Devnagari Handwritten Character Recognition using Different Feature and Classifiers”, 10th Intl. Conf. on Document Analysis and Recognition, pp. 1111-1115, 2009.

[41] H. Freeman, “On the encoding of arbitrary geometric configurations”, *IRE Transactions on Electronic Computers EC*, 10(1961) : 260-268.

[42] L. Miclet, “Méthodes structurelles pour la reconnaissance des formes”, Ed, Eyrolles, collection technique et scientifique des télécommunications, 1984.

[43] X. Gao, B. Xiao, D. Tao, X. Li, A survey of graph edit distance, Pattern Anal. Appl. 13 (2010) 113–129.

[44] H. Bunke, C. Irniger, M. Neuhaus, “Graph matching - challenges and potential solutions”, in: Image Analysis and Processing, Lecture Notes in Computer Science, vol. 3617, Springer, pp.1–10, 2005.

[45] C. L. Liu, B. Luo, and W. Kropatsch. “Advances in graph-based pattern recognition” editorial. 2016.

[46] H. Bunke, K. Riesen, “Towards the unification of structural and statistical pattern recognition”, Pattern Recognit. Lett. 33, 811–825, 2012.

[47] R. C. Wilson, P. Zhu, “A study of graph spectra for comparing graphs and trees”, Pattern Recognit. 41, 2833–2841, 2008.

[48] T. Gaertner, J. W. Lloyd, P. A. Flach, “Kernels for Structured Data”, Springer, 2003.

[49] D. Conte, P. Foggia, C. Sansone, M. Vento, “Thirty years of graph matching in Pattern Recognition”, Int. J. Pattern Recognit. Artif. Intell. 18, 265–298,2004.

[50] P. Foggia, G. Percannella, M. Vento, “Graph matching and learning in pattern recognition in the last ten years”, Int. J. Pattern Recognit. Artif. Intell., inpress. Doi : http://dx.doi.org/10.1142/S0218001414500013), 2015.

[51] A. Belaïd et Y. Belaïd, "Reconnaissance des formes, méthodes et applications", Inter éditions, 1992.

[52] B. Alsallakh, H. Safadi, “AraPen : an Arabic online handwriting recognition system”, Proc. of ICTTA’06, 2nd IEEE International Conference on Information & Communication Technologies : from Theory to Applications, vol.1, pp. 1844-1849, Damascus, Syria, April 2006.

[53] F. Menasri, “Contributions à la reconnaissance de l’écriture arabe manuscrite”, Thèse de doctorat, Université paris Descartes, 2008.

[54] R. EL-Hajj, L. Likforman-Sulem, C. Mokbel, “Combining slanted-frame classifiers for improved HMM-based Arabic handwriting recognition”, IEEE PAMI, vol.31 (7), pp 1165-1177, 2009.

[55] H. Zouari, “Contribution à l’évaluation des méthodes de combinaison parallèle de classifieurs par simulation”, Thèse de Doctorat, Université de Rouen, 2004.

[56] A. Oulamara, J Duvernoy, “An application of the Hough transform to automatic recognition of Berber characters”, Signal Processing, vol.14, pp.79- 90, 1988.

[57] Djematen, B. Taconet, A. Zahour: “Une méthode statistique pour la reconnaissance de caractères berbères manuscrits”, *CIFED’98*, pp.170-178, 1998.

[58] Y. Ait Ouguengay, M. Taalabi, “Elaboration d’un réseau de neurones artificiels pour la reconnaissance optique de la graphie Amazighe: Phase d’apprentissage”, *Systèmes intelligents-Théories et applications, Paris : Europia, cop. 2009 (impr. au Maroc)*, ISBN-102909285553, 2009.

[59] M. Amrouch, Y. Es Saady, A. Rachidi, M. El Yassa, D. Mammass, “Printed Amazigh Character Recognition by a Hybrid Approach Based on Hidden Markov Models and the Hough Transform”, *ICMCS’09*, Ouarzazate, 2009.

[60] M. Amrouch, Y. Es-Saady, A. Rachidi, M. El-Yassa, D. Mammass, “A novel feature set for recognition of printed amazigh text using maximum deviation and hmm”. *Int. J. Comput. Appl*, 44, 2012.

[61] R. El Ayachi, K. Moro, M. Fakir, B. Bouikhalene, “On the Recognition of Tifinaghe Scripts”, *Journal of Theoretical and Applied Information Technology*, 20(2) :61-66, 2010.

[62] R. EL Ayachi, M. Fakir and B. Bouikhalene, “Recognition of Tifinaghe Characters Using a Multilayer Neural Network”, *International Journal Of Image Processing (IJIP)*, 5(2), 2011.

[63] R El Ayachi, M Oujaoura, M Fakir and B Minaoui. “Code Braille et la reconnaissance d’un document écrit en Tifinagh”, *The International Conference on Information and Communication Technologies for the Amazigh (TICAM)*, 2014.

[64] Y. Es Saady, A. Rachidi, M. El Yassa, D. Mammass, “Printed Amazigh Character Recognition by a Syntactic Approach using Finite Automata”, *ICGST-GVIP Journal*, vol.10, Issue 2, pp.1-8, 2010.

[65] Y. Es Saady, A. Rachidi, M. El Yassa, D. Mammass, “Une méthode syntaxique pour la reconnaissance de caractères amazighs imprimés”, *CARI’08*, Maroc, 27-31 Octobre 2008.

[66] O Bencharef, M Fakir, B Minaoui and B Bouikhalene, Tifinagh Character Recognition Using Geodesic Distances, Decision Trees & Neural Networks. *(IJACSA) International Journal of Advanced Computer Science and Applications, Special Issue on Artificial Intelligence*, 2011.

[67] M. Oujaoura, et al. "Invariant descriptors and classifiers combination for recognition of isolated printed Tifinagh characters." *Third international symposium on Automatic Amazigh processing (SITACAM'13). (2013) Beni-Mellal, Morocco.*

[68] M. Oujaoura, R. El Ayachi, B. Minaoui, M. Fakir, B. Bouikhalene, O. Bencharef, Invariant descriptors and classifiers combination for recognition of isolated printed Tifinagh characters. *In International Journal of Advanced Computer Science and Application, Special Issue on Selected Papers from Third international symposium on Automatic Amazigh processing* (pp. 22-28), 2013.

[69] M. Amrouch, A. Rachidi, M. El Yassa, D. Mammass. "Handwritten Amazigh Character Recognition Based On Hidden Markov Models". *ICGST-GVIP Journal*, 10(5), 2010.

[70] B. El Kessab, C. Daoui, K. Moro, B. Bouikhalene, M. Fakir. "Recognition of Handwritten Tifinagh Characters Using a Multilayer Neural Networks and Hidden Markov Model". *Global Journal of Computer Science and Technology*, 11(15), 2011.

[71] S. Gounane and M. Fakir, B. Bouikhalen. "Handwritten Tifinagh Text Recognition Using Fuzzy K-NN and Bi-gram Language Model". *IJACSA Special Issue on Selected Papers from Third international symposium on Automatic Amazigh processing (SITACAM' 13)*, 2013.

[72] Y. Es-Saady, M. Amrouch, A. Rachidi, M. El Yassa, D. Mammass. "Handwritten Tifinagh Recognition Using Baselines Detection Features". *International Journal of Scientific & Engineering Research*, 5(4), April-2014.

[73] H. Bunke, A. Sanfeliu, "Syntactic and Structural Pattern Recognition: Theory and Applications", vol. 7, World Scientific, 1990.

[74] P. Foggia, R. Genna, M. Vento, "Symbolic vs. connectionist learning: an experimental comparison in a structured domain", IEEE Trans. Knowl. Data Eng. 13 (2001) 176–195.

[75] L. P. Cordella, P. Foggia, C. Sansone, M. Vento, "Learning structural shape descriptions from examples", Pattern Recognit. Lett. 23 (2002) 1427–1437.

[76] A. V. Aho, J. E. Hopcroft, J. D. Ullman. "The Design and Analysis of Computer Algorithms", Addison-Wesley Publishing Company, 1974.

[77] J. E. Hopcroft, J. K. Wong, "Linear time algorithm for isomorphism of planar graphs (preliminary report)", in: Annual ACM Symposium on Theory of Computing", ACM, New York, USA, pp. 172–184, 1974.

[78] J. R. Ullmann, "An algorithm for subgraph isomorphism", J. ACM 23, 31–42,1976.

[79] D. C. Schmidt, L. E. Druffel, “A fast backtracking algorithm to test directed graphs for isomorphism using distance matrices”, J. ACM 23, 433–445, 1976.

[80] D. E. Ghahraman, A. K. C. Wong, T. Au, Graph optimal monomorphism algo-rithms, IEEE Trans. Syst. Man Cybern. 10 (1980) 181–188.

[81] L. P. Cordella, P. Foggia, C. Sansone, F. Tortorella, M. Vento, “Graph matching: a fast algorithm and its evaluation”, in: ICPR, vol. 2, pp. 1582–1584, 1998.

[82] M. De Santo, P. Foggia, C. Sansone, M. Vento, “A large database of graphs and its use for benchmarking graph isomorphism algorithms”, Pattern Recognit. Lett. 24, 1067–1079, 2003.

[83] L. P. Cordella, P. Foggia, C. Sansone, M. Vento, “A (sub)graph isomorphism algorithm for matching large graphs”, IEEE Trans. Pattern Anal. Mach. Intell. 26, 1367–1372, 2004.

[84] J. Larrosa, G. Valiente, “Constraint satisfaction algorithms for graph pattern matching”, Math. Struct. Comput. Sci. 12, 403–422, 2002.

[85] J.R. Ullmann, Bit-vector algorithms for binary constraint satisfaction and subgraph isomorphism, J. Exp. Algorithm. 15, 1.6:1–1.6:64, 2011.

[86] S. Zampelli, Y. Deville, C. Solnon, Solving subgraph isomorphism problems with constraint programming, Constraints 15, 327–353, 2010.

[87] C. Solnon, AllDifferent-based filtering for subgraph isomorphism, Artif. Intell. 174 (2010) 850–864.

[88] B. D. McKay, “Practical Graph Isomorphism”, 1981.

[89] P. Hart, N. Nilsson, B. Raphael, “A formal basis for the heuristic determination of minimum cost paths”, IEEE Trans. Syst. Sci. Cybern. 4, 100–107, 1968.

[90] W. H. Tsai, K. Fu, “Error-correcting isomorphisms of attributed relational graphs for pattern analysis”, IEEE Trans. Pattern Anal. Mach. Intell. 9, 757–768, 1979.

[91] W. H. Tsai, K. Fu, “Subgraph error-correcting isomorphisms for syntactic pattern recognition”, IEEE Trans. Syst. Man Cybern. 13, 48–62, 1983.

[92] A. K. C. Wong, M. You, S. C. Chan, “An algorithm for graph optimal monomorph-ism”, IEEE Trans. Syst. Man Cybern. 20, 628–638, 1990.

[93] A. Sanfeliu, K. S. Fu, “A distance measure between attributed relational graphs for pattern recognition”, IEEE Trans. Syst. Man Cybern, 353–362, 1993.

[94] K. Riesen, H. Bunke, “Approximate graph edit distance computation by means of bipartite graph matching”, Image Vis. Comput. 27, 950–959, 2009.

[95] L.P. Cordella, P. Foggia, C. Sansone, M. Vento, “Subgraph transformations for the inexact matching of attributed relational graphs”, Computing 12, 43–52, 1998.

[96] F. Serratosa, R. Alquezar, A. Sanfeliu, “Function-described graphs: a fast algorithm to compute a sub-optimal matching measure”, in: Graph-Based Representations in Pattern Recognition, pp. 71–77, 1999.

[97] F. Serratosa, R. Alquezar, A. Sanfeliu, “Efficient algorithms for matching attributed graphs and function-described graphs”, in: ICPR, pp. 867–872, 2000.

[98] M.A. Fischler, R.A. Elschlager, “The representation and matching of pictorial structures”, IEEE Trans. Comput. 22, 67–92, 1973.

[99] H. A. Almohamad, S. O. Duffuaa, “A linear programming approach for the weighted graph matching problem”, IEEE Trans. Pattern Anal. Mach. Intell. 15, 522–525, 1993.

[100] A. Rangarajan, E. Mjolsness, A Lagrangian relaxation network for graph matching, in: IEEE International Conference on Neural Networks, vol. 7, pp. 4629–4634, 1994.

[101] S. Gold, A. Rangarajan, “A graduated assignment algorithm for graph matching”, IEEE Trans. Pattern Anal. Mach. Intell. 18, 377–388, 1996.

[102] S. Umeyama, “An Eigen decomposition approach to weighted graph matching problems”, IEEE Trans. Pattern Anal. Mach. Intell. 10 (1988) 695–703.

[103] M. Carcassoni, E. R. Hancock, “Weighted graph-matching using modal clusters, in: CAIP”, Springer-Verlag, London, UK, pp. 142–151, 2001.

[104] S. Kosinov, T. Caelli, “Inexact multisubgraph matching using graph eigenspace and clustering models”, in: S+SSPR, Springer-Verlag, London, UK, pp. 133–142, 2002.

[105] X. Jiang, A. Munger, H. Bunke, “A median graphs: properties, algorithms, and applications”, IEEE Trans. Pattern Anal. Mach. Intell. 23, 1144-1151, 2001.

[106] R. O. Duda, P. E. Hart, D. G. Stork, “Pattern Classification”, 2nd edition, Wiley, New York, 2001.

[107] S. Günter, H. Bunke, Self-organizing map for clustering in the graph domain, Pattern Recognit. Lett. 23, 405–417, 2002.

[108] M. Neuhaus, H. Bunke, “Bridging the Gap between Graph Edit Distance and Kernel Machines”, World Scientific, River Edge, USA, 2007.

[109] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, K. Borgwardt, Efficient Graphlet Kernels for Large Graph Comparison, in: International Conference on Artificial

Intelligence and Statistics, Society for Artificial Intelli-gence and Statistics, Clearwater Beach, USA, 2009.

[110] P. Mahe, J. P. Vert, “Graph kernels based on tree patterns for molecules”, Mach. Learn. 75, 3–35, 2009.

[111] N. Shervashidze, K. Borgwardt, in: Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, A. Culotta (Eds.), “Advances in Neural Information Processing Systems”, vol. 22, Neural Information Processing Systems Foundation, pp. 1660–1668, 2009.

[112] H. Kashima, K. Tsuda, A. Inokuchi, “Marginalized kernels between labeled graphs, in: International Conference on Machine Learning”, AAAI Press, pp. 321–328, 2003.

[113] F. X. Dupé, L. Brun, “Tree covering within a graph kernel framework for shape classification”, in: ICIAP, pp. 278–287, 2009.

[114] B. Luo, R. C. Wilson, E. R. Hancock, “Spectral embedding of graphs”, Pattern Recognit. 36, 2213–2230, 2003.

[115] R. Wilson, E. Hancock, B. Luo, “Pattern vectors from algebraic graph theory”, IEEE Trans. Pattern Anal. Mach. Intell. 27, 1112–1124, 2005.

[116] X. Bai, H. Yu, E. Hancock, “Graph matching using spectral embedding and alignment”, in: ICPR, vol. 3, pp. 398–401, 2004.

[117] A. Torsello, E. R. Hancock, “Graph embedding using tree edit-union”, Pattern Recognit. 40, 2007, 1393–1405.

[118] D. Emms, R. C. Wilson, E. Hancock, “Graph embedding using quantum commute times”, in: Graph-Based Representations in Pattern Recognition, Springer-Verlag, pp. 371–382, 2007.

[119] K. Riesen, M. Neuhaus, H. Bunke, “Graph embedding in vector spaces by means of prototype selection”, in: Graph-Based Representations in Pattern Recognition, *Springer-Verlag*, pp. 383–393, 2007.

[120] T Y Zhang, C Y Suen. “À Fast Parallel Algorithm for Thinning Digital Patterns”. *Communications of the ACM*. 27(3), 1984.

[121] M Leordeanu, M Hebert. “A spectral technique for correspondence problems using pairwise constraints”, *International Conference of Computer Vision (ICCV'05),* 2 : 1482-1489). IEEE, 2005.

[122] Y. Es Saady, A. Rachidi, M. El Yassa and D. Mammass: "AMHCD: A Database for Amazigh Handwritten Character Recognition Research", *International Journal of Computer Applications*, 27(4):44-48, August 2011.

[123] W. J. Krzanowski, Principles of Multivariate Analysis: A User's Perspective. *New York: Oxford University Press*, 1988.