THÈSE

pour l'obtention du Grade de
DOCTEUR DE L'UNIVERSITÉ DE POITIERS
(Université de Poitiers)
(Diplôme National - Arrêté du 25 Avril 2002)

École Doctorale : **Sciences Pour l'Ingénieur et Aéronautique**
Secteur de Recherche : **Traitement du Signal et des Images**

Présentée par :
Denis ARRIVAULT

Apport des Graphes dans la Reconnaissance Non-Contrainte de Caractères Manuscrits Anciens

Directrice de Thèse :
Christine FERNANDEZ-MALOIGNE

Soutenue le 17 Mars 2006 devant la commission d'examen composée de :

N. Vincent, Professeure, Université Paris VI, CRIP5 ..... Rapporteur
W.G. Kropatsch, Professeur, Université de Leibnitz (Autriche), PRIP ..... Rapporteur

I. Bloch, Professeure, Ecole Nationale Supérieure des Télécommunications, LTCI ..... Examinateur
J.M. Ogier, Professeur, Université de La Rochelle, L3i ..... Examinateur

N. Richard, Maître de Conférences, Université de Poitiers, SIC ..... Co-directeur de Thèse

C. Fernandez-Maloigne, Professeure, Université de Poitiers, SIC ..... Directrice de Thèse

P. Bouyer, Directeur Technique, Société RC-Soft, Saint Yrieix ..... Invité

REMERCIEMENTS

Remercier est une tâche ingrate puisqu'elle conduit inévitablement à des oublis. Je vais donc commencer par remercier tous ceux qui de près ou de loin ont contribué par un mot, un geste, une attention à ce travail et qui ne seront pas cités par la suite.

Je tiens, ensuite, à exprimer toute ma gratitude à NICOLE VINCENT et WALTER G. KROPATSCH pour avoir accepté de rapporter sur ce travail de thèse. Merci pour tous vos commentaires et vos questions qui ont enrichi ce mémoire. Un grand merci à ISABELLE BLOCH qui a accepté d'être présidente du jury ainsi qu'à JEAN-MARC OGIER qui en a été examinateur. Merci pour vos remarques et votre enthousiasme.

Je voulais, ensuite, remercier tout particulièrement trois personnes sans qui ce travail n'aurait jamais pu voir le jour : CHRISTINE FERNANDEZ-MALOIGNE qui a dirigé cette thèse, NOËL RICHARD qui m'a encadré et épaulé avec ferveur et passion et PHILIPPE BOUYER qui est l'initiateur du projet COROC, le co-fondateur de l'entreprise RC-Soft, qui a financé cette thèse, et surtout un industriel qui investit son temps et son énergie pour développer des projets d'innovation. Merci d'avoir supporté mes très rares ronchonnements et de m'avoir soutenu jusqu'au bout.

Merci à toute l'équipe de RC-Soft et particulièrement à STÉPHANE SOCHAKI et NICOLAS LARCHER pour tous les moments et les idées partagés. Merci à PATRICK TRIAUD de TDI-Services pour avoir soutenu ce projet.

Merci à SYLVAIN CHARNEAU et à GUILLAUME LEBRUN pour votre aide et votre travail. Vous avez tous les deux abandonné les hiéroglyphes après votre stage, l'Egypte Ancienne y a certainement perdu beaucoup.

Je voulais également remercier toutes les personnes du laboratoire SIC dont beaucoup sont maintenant des amis. Merci à JULIEN DOMBRE pour une certaine soirée parisienne et également pour ton aide et ton travail. Merci à BENOÎT TREMBLAIS et GUILLAUME DAMIAND pour tous les échanges enrichissants que nous avons pu avoir et qui ont énormément contribué à ce travail. Merci à PHILIPPE CARRÉ, à BERTRAND AUGEREAU et à toute l'équipe ICONE qui grandit, qui grandit... Merci à FRANÇOISE PERRAIN pour ta bonne humeur et ton dévouement. Merci à Fred, Sam, Pascal et tous les thésards de la cafet'. Merci à PHILIPPE DUBOIS pour ta science des posters. Merci à tous mes collègues de bureau pour m'avoir supporté : Anne-Sophie, Chaker, Alice, Benoît, Julien, Nicolas, Carine et Michèle.

Enfin et surtout, je voulais remercier toutes ces rencontres poitevines qui ont fait de cette thèse, un moment riche et humain. Merci à vous So & Yannick, Christelle & Guillaume, Fred & David, Benjamin, Aline & Benoît, Marion & Fabrice, Julien, François, Michtouille, Christian, Bertrand, Sylvie & Philippe, Isabelle & Laurent, Gaëlle & Xavier ainsi que toute l'équipe théâtrale de Nouaillé de ces quatre dernières années...

A Aurélie, Maud et Gérard pour toutes ces années passées à vos côtés,
à toi Sébastien du Ch'nord.

A Denise partie trop tôt et à Robert pour l'amour et le sens.

« Mille choses douces sans nom
Qu'on fait plus qu'on ne les remarque
Mille nuances d'êtres humaines
A demi-songe à demi-joie
Rien moins que rien pourtant la vie »

Louis Aragon

SOMMAIRE

CHAPITRE 1

INTRODUCTION

Sommaire

« [...] le langage sert aussi à communiquer des idées. Les mots interviennent alors par leur signification, qui est l'idée que l'on a de ce qu'ils désignent. Les nominalistes disent qu'il n'y a pas d'idées universelles, mais seulement ce que moi je pense de ce que les mots désignent. Peu importe alors que les ordinateurs ne pensent rien de ces désignations, c'est une question qui leur est propre. Tout le langage doit pouvoir être traité par ordinateur, aux restrictions précédentes près. Une intelligence artificielle est possible. Les réalistes disent qu'il y a des idées universelles : ce n'est pas moi qui décide ce qu'est l'homme, l'idée existe hors de moi, il m'appartient d'essayer de la découvrir. L'ordinateur n'ayant pas accès à ces idées se trouve en position d'infériorité, le langage symbolique lui échappe, une intelligence artificielle est impossible. On peut croire que l'expérience départagera les deux clans. Mais il faut rappeler qu'il ne suffira pas que quelques textes soient correctement traités, il faudra montrer que tous peuvent l'être. On imagine difficilement comment cela pourrait se faire. La question du sens posée par l'informatique est la même que celle de l'existence d'idées universelles, ou celle de l'existence de la vérité. Il y a plus de vingt-cinq siècles qu'on en débat, sans avoir pu la trancher. Je suis intimement convaincu qu'elle ne sera pas tranchée par l'informatique, elle me paraît indécidable par l'esprit humain. »

JACQUES ARSAC.

« L'informatique pose la question du sens »[1],[Ars03].

La reconnaissance automatique de caractères, au sens large du terme, est une discipline vieille comme l'ordinateur. Elle pose la question de l'intelligence humaine et des possibilités de l'intelligence artificielle. Pourquoi nous semble-t-il si facile de déchiffrer un texte manuscrit (même relativement mal écrit)

[1] http ://www.asmp.fr/travaux/gpw/philosc/rapport3/5arsac.pdf

alors même que cette tâche est si complexe à automatiser ?

La LAD[2] et plus généralement la RdF[3] sont des domaines de recherche actifs depuis la fin des années soixante ([Rut66]). Des sytèmes ont vu le jour pour des applications très spécifiques, mais comparés aux fantastiques progrès réalisés en médecine par exemple, et malgré une recrudescence des travaux de recherche depuis les années 80, les avancées restent lentes et encore décevantes. Pourtant le champ de la RdF est considérable et les publications scientifiques dans ce domaine sont nombreuses. Elles atteignaient déjà le millier par an au début des années 80 [Sim84] et sont en explosion depuis. De plus, d'un point de vue mathématique, le problème posé par la RdF est trivial, comme le soulignait SIMON en 1984 [Sim84] : « Soit X un espace de représentation - de préférence un « bon espace topologique » - et Ω un ensemble fini de noms -l'espace d'interprétation-. Une reconnaissance - une identification - est une application $\mathfrak{E} : X \rightarrow \Omega$, que l'on va pourvoir de « propriétés » ; on en déduira de jolis théorèmes... ». Alors où se situe la difficulté ?

SIMON, dans ce même ouvrage, la place sur deux niveaux. La première difficulté réside dans la complexité calculatoire. En effet la construction d'un espace de représentation pour une application donnée, et plus particulièrement pour la reconnaissance de caractères manuscrits, se heurte hier, comme aujourd'hui, à des problèmes informatiques de temps de calcul, de temps d'accès ou de stockage. La seconde difficulté se situe dans la sémantique même du problème ;

- Existe-t-il une technique générale pour construire un opérateur de RdF ?
- Peut-on construire un système auto-apprenant et auto-organisateur capable de reconnaître n'importe quelle forme pour peu qu'elle ait été apprise ?

Le problème de la complexité informatique, bien que toujours d'actualité et au cœur de la recherche, tend à être relégué au second rang grâce aux formidables progrès technologiques réalisés depuis les années 80 et à tous les travaux d'optimisation algorithmique. C'est pourquoi, et c'est la motivation de ce travail de thèse, il est intéressant de regarder aujourd'hui du côté des modélisations dites complexes pour tenter de répondre à la deuxième difficulté énoncée par SIMON.

Commençons par survoler l'historique de la LAD et plus spécifiquement de la reconnaissance d'écriture. Nous ne reviendrons pas sur les débuts de l'informatique dont les capacités de stockage et de représentation ne permettent pas de vrais traitements optiques des textes. Tout commence véritablement vers le milieu des années 60 avec la création du Extended ASCII (American Standard Code for Information Interchange) en 1965 qui permet la représentation de 256 caractères. Ensuite il y a, grossièrement, trois périodes distinctes dans le développement de l'OCR[4] ([AYV01]).

1960-1980 ; Les premiers âges : Avec l'apparition des premiers ordinateurs se développent les premiers systèmes OCR dans les années 60 (Citons entre autres celui du centre de recherche Thomas J. Watson d'IBM à New-York [KL63], [Liu64], [Pot64], [NS66] ou les travaux du Bell Telephone Labor [Har72]). Ces systèmes fonctionnent essentiellement sur des caractères typographiés et utilisent des techniques dites de *Template Matching*, qui comparent les images pixel à pixel à l'aide de classifieurs statistiques. Ils sont dédiés aux caractères latins et numériques. Quelques études voient également le jour sur des écritures plus complexes principalement les langues asiatiques (japonnais, chinois et coréen [NC66], [Sta72], [MYY84], [HL93]) et l'arabe ([AKHM80]).

Parallèlement, des systèmes commerciaux de reconnaissance en-ligne (sur tablette) font leur apparition. Au Rand Laboratories de Santa Monica en Californie, la démonstration du fonctionnement est effectuée, sans trop de succès commercial, avec la Rand Tablet [DE64], figure 1.1 page ci-contre. Elle permet à un usager d'écrire à la main et de voir son écriture reconnue par l'ordinateur puis transformée

[2] Lecture Automatique de Document
[3] Reconnaissance de Formes
[4] Optical Character Recognition

en caractères imprimés.

FIGURE 1.1 – Rand Tablet ou Grafacon

1980-1990 ; Développements : Cette période est marquée par l'arrivée de la puissance de calcul et de stockage. Elle entraîne logiquement un développement accru des systèmes d'OCR ([Sue86], [Man86]). Les travaux concernent chaque étage de la chaîne de traitements, de la segmentation à la reconnaissance, de façon relativement indépendante. Les approches structurelles à base de grammaires syntaxiques commencent également à voir le jour ([BH84]).

Après 1990 ; Avancées : Depuis le début des années 1990, nous observons une augmentation importante des travaux en reconnaissance d'écriture. Les barrières technologiques ne cessent d'être franchies et les systèmes présentés sont de plus en plus complexes. Des techniques modernes d'intelligence artificielle, utilisées seules ou combinées, donnent de très bons résultats : les réseaux de neurones, les modèles de Markov cachés, les raisonnements flous ([LG02], [BS97], [SA99], [MP00], [KLSS02], [PI98], [SMle02], [KKS00])... Pour ce qui est, par exemple, de la reconnaissance d'adresses manuscrites ou de montants de chèques, des logiciels commerciaux sont maintenant disponibles ([GAA$^+$01], [KB00], [LLG95], [KHP05]). Néanmoins nous sommes encore loin du système de reconnaissance générique ou non contraint. Tous les systèmes nécessitent un apprentissage conséquent sur un domaine d'application restreint.

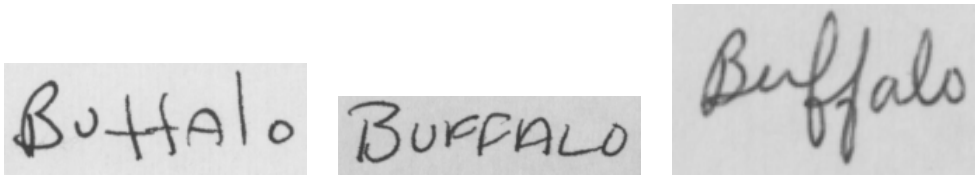
1.1 Problématique de la Reconnaissance Automatique de l'Ecriture et du Document ou RAED

Il est d'usage de distinguer deux activités en RAED, les reconnaissances *en-ligne* et *hors-ligne*. La reconnaissance en-ligne s'effectue à partir d'une acquisition spatio-temporelle des caractères ou mots cursifs sur une tablette électronique (version moderne de la Rand Tablet présentée sur la figure 1.1). L'article de TAPPERT & AL donne un bon inventaire des méthodes utilisées en reconnaissance en-ligne ([TSW90]). La reconnaissance hors-ligne, privée de l'information spatio-temporelle, est plus délicate. Elle concerne, *a priori*, tous les documents écrits puis numérisés (adresses postales, chèques banquaires, formulaires structurés ou manuscrits quelconques). Bien que ces deux domaines d'application soient fortement liés, nous ne nous intéresserons dans ce mémoire qu'à la reconnaissance hors-ligne.

D'une manière générale la complexité d'un système de RAED s'évalue suivant trois critères orthogonaux ([Bel01]) :

– *Disposition spatiale du texte :* la présentation d'un texte varie globalement entre deux formats : l'écriture *contrainte* correspondant à une écriture guidée par des cadres (les formulaires par exemple) et l'écriture *non-contrainte* correspondant à une écriture guidée exclusivement par le scripteur donc extrêmement variable. Les écritures externes ou internes détachées (écriture en bâtons) sont,

bien entendu, les plus aisées à traiter du fait de la séparation plus ou moins immédiate des lettres (figure 1.2, extraits de la Base CEDAR[5]).



(a) (b) (c)

FIGURE 1.2 – Différents types d'écritures : (a) écriture en bâtons, (b) écriture majuscule, (c) écriture attachée.

- *Nombre de scripteurs :* la difficulté de traitement croît avec le nombre de scripteurs. Trois catégories d'écritures se distinguent : les écritures *monoscripteurs*, *multiscripteurs* et *omniscripteurs*. En mode multiscripteurs, le système doit être capable de reconnaître l'écriture de plusieurs personnes prédéfinies, alors qu'en omniscripteur il doit s'adapter à n'importe qui.
- *Taille du vocabulaire :* Il existe deux types d'applications, celles qui sont à vocabulaire limité (< 100 mots) et celles qui sont à vocabulaire très étendu (> 10 000 mots). La reconnaissance sera bien plus aisée dans le premier cas puisqu'il sera possible de comparer un mot inconnu avec la totalité des mots du dictionnaire. Dans le second cas il faudra bien souvent mettre en oeuvre des décisions hiérarchiques pour réduire le temps de calcul et l'encombrement mémoire.

La figure 1.3 présente un shéma synthétique de la complexité des systèmes RAED ([Bel01]). Plus l'application s'éloigne du centre du repère, plus la reconnaissance devient difficile.

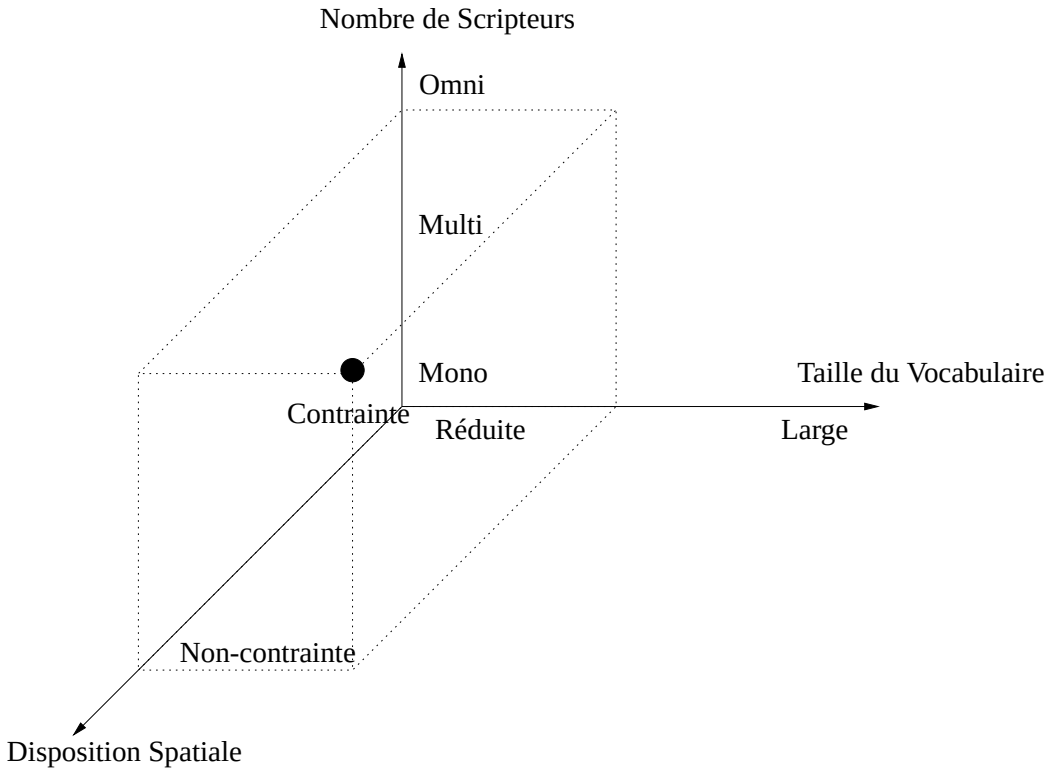


FIGURE 1.3 – Graphe de complexité des systèmes de RAED d'après BELAID [Bel01]

[5]Center of Excellence for Document Analysis and Recognition (http ://www.cedar.buffalo.edu/)

Aujourd'hui les systèmes d'OCR contraints et à vocabulaire restreint donnent des résultats étonnants. Les taux de reconnaissance affichés par ce genre de logiciel avoisinent les 100% pour les documents imprimés ou mono-scripteurs de bonne qualité. Bien souvent, ils fonctionnent suivant un schéma classique représenté sur la figure 1.4. Le texte d'apprentissage et le texte à reconnaître sont homogènes (présegmentés en composantes primitives, caractères, mots ou phrases) et la phase de prétraitement dépend de la nature des documents (elle peut aller du débruitage à la segmentation).

Ce processus, décrit de manière séquentielle, accepte également des rebouclages, soit sur des critères calculés à la fin de chacune des étapes, soit directement en sortie de classification suivant un indice de confiance dépendant du ou des classifieurs utilisés. De plus, ce processus se spécialise suivant deux applications en fonction du type de composante primitive considérée. Pour les écritures latines, qui restent les écritures les plus étudiées, la reconnaissance se fait soit au niveau du caractère, soit au niveau du mot ([BBOM05], [Bun03]). Nous ne parlons pas ici des techniques plus générales d'analyse de documents qui visent à extraire la structure d'un document en vue d'une indexation sur son type ([TCL+98]).

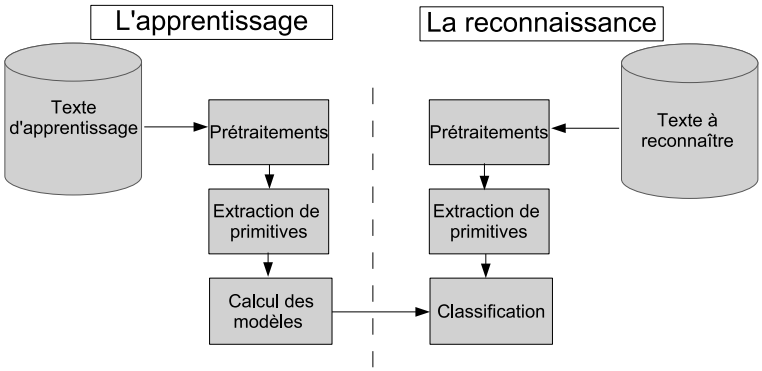


FIGURE 1.4 – Schéma global d'un processus de reconnaissance de caractère

1.1.1 Reconnaissance de caractères

C'est la partie la plus étudiée des systèmes de reconnaissance d'écrits car la plus ancienne et la plus simple quand il s'agit de caractères numériques ou latins (vocabulaire limité). Le dictionnaire est constitué d'images de caractères étiquetées. Une image de caractère inconnu pourra être étiquetée comme l'élément du dictionnaire dont elle est la plus similaire. La notion de similarité implique une description des caractères dans un espace de représentation muni d'une métrique. Il existe une multitude de descripteurs dits de forme, nous distinguerons les descriptions vectorielles dites *statistiques* des descriptions syntaxiques et/ou topologiques des formes dites *structurelles*.

Dans le cas d'une description statistique de la forme, le problème est ramené à un exercice d'analyse de données classique. La classification est basée sur une distance (arbres de décisions, k plus proches voisins...) ([JDM00]) et peut être de nature probabiliste ou neuronale.

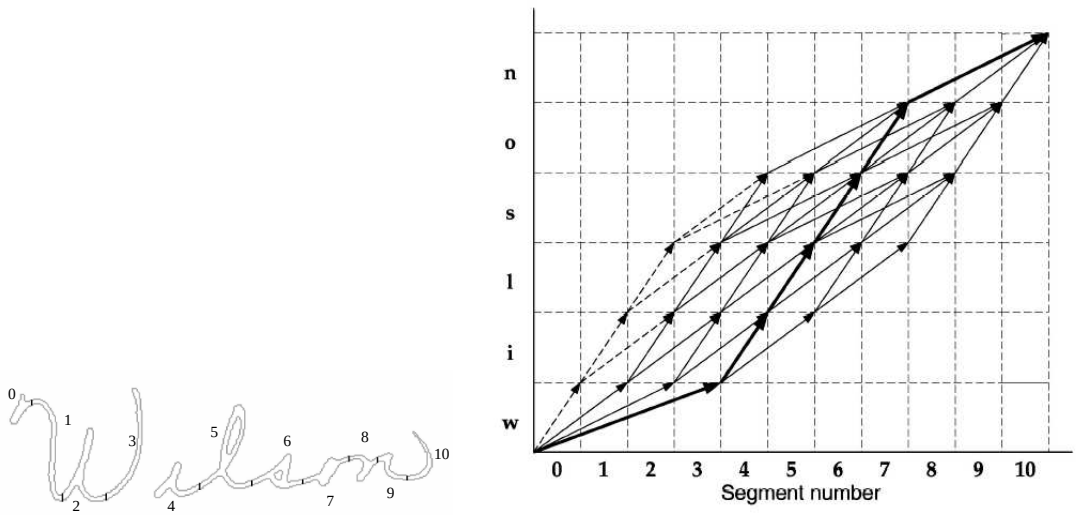
Dans le cas d'une description structurelle, il faudra utiliser des formalismes plus complexes. Les formes sont décomposées en primitives simples qui peuvent être des graphèmes ou même les pixels de l'image. Elles sont ensuite représentées par un objet complexe, composé des primitives, comme une chaîne ou un graphe. Le processus de reconnaissance (grammatical [GT78], stochastique [BS97] ou graphique [Bun01]) est propre à la représentation utilisée.

1.1.2 Reconnaissance de mots

Il y a trois façons d'aborder cette problématique [Vin02]. Soit le système reconnaît le mot comme une entité entière et indivisible, il s'agit d'une approche *globale* ou *holistique*. Soit il reconnaît le mot à partir de ses caractères préalablement segmentés, il s'agit d'une approche *analytique*. Soit il n'utilise que certaines propriétés et raffine sa description du mot par rebouclage, nous parlerons alors de *systèmes basés sur la lecture humaine*.

Les approches globales reposent sur une description du mot du même type que celles utilisées en reconnaissance de caractères. Comme le mot est une forme plus complexe que le caractère, la description contiendra plus d'informations et sera ainsi moins sensible aux changements de scripteurs. Néanmoins le vocabulaire étant plus large, l'apprentissage nécessite beaucoup d'individus ce qui rend, en pratique, cette approche inutilisable. Pour cette raison, les approches globales sont souvent utilisées pour d'autres tâches. Elles servent, par exemple, à réduire la taille du lexique ([MK97]) ou à valider une reconnaissance analytique ([CHS94]).

Les approches analytiques reposent sur l'identification de parties de l'image du mot comme étant des parties de l'ensemble des modèles prédéfinis et connus du classifieur. La difficulté de cette approche a été explicitée par SAYRE [Say73] : « pour reconnaître les lettres, il faut segmenter le tracé et pour segmenter le tracé, il faut reconnaître les lettres ». Il convient donc d'adopter une méthode itérative qui raffine la segmentation en fonction des hypothèses de classification. Il y a deux façons de résoudre ce problème. La première consiste à « disséquer » ([CL96]) le mot en graphèmes et utiliser la reconnaissance de ces graphèmes pour corriger les erreurs de segmentation (figure 1.5). La seconde consiste à scanner l'image du mot pour y reconnaître des lettres puis à utiliser un dictionnaire afin de valider la reconnaissance (figure 1.6 page ci-contre).



(a) (b)

FIGURE 1.5 – Reconnaissance analytique de mots basée graphème, (a) dissection, (b) schéma de la reconnaissance du mot WILSON par appariement dynamique des segments ou graphèmes ([MK97]).

Les systèmes basés sur la lecture humaine reposent sur le principe de la supériorité du mot (*word superiority effect*, [Rei69]). Ce principe veut qu'une lettre soit plus facile à reconnaître dans un mot que seule. Un effet secondaire de ce principe, que l'on a tous expérimenté, est la capacité humaine de reconnaître un mot alors même que quelques unes de ses lettres sont inversées. Il apparaît donc que la perception de formes particulières dans un mot suffisent à sa lecture. La figure 1.7 page suivante donne

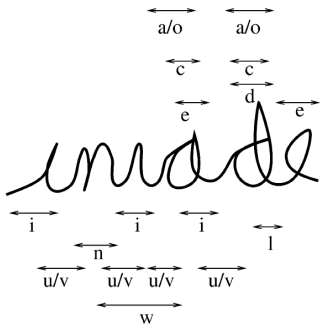


FIGURE 1.6 – Reconnaissance analytique de mots basée sur la reconnaissance de lettres ([MK97]).

un exemple de description simple, basée sur les boucles, les hampes et les jambages. Tous les mots se représentent de manières différentes excepté « five » et « four » qui présentent une ambiguïté nécessitant un raffinement dans la détection de primitives.

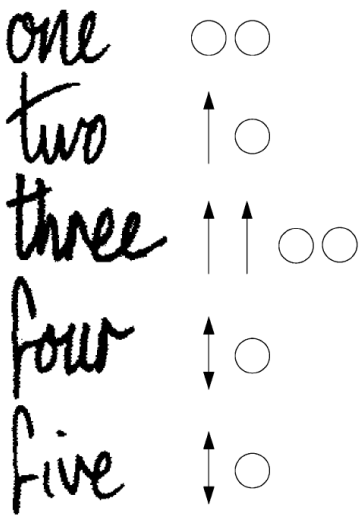


FIGURE 1.7 – Exemple de reconnaissance basée sur la lecture humaine ([Vin02]).

1.2 Du grec au hiéroglyphe, la difficulté des systèmes RAED multilingues

La RAED sur les écritures complexes, omni-scripteurs à large vocabulaire et non-contraintes, n'est toujours pas à l'ordre du jour, et il semble que le système auto-apprenant et auto-organisateur capable de reconnaître n'importe quelle forme de J.C. SIMON ne soit pas envisageable dans les années à venir. Cependant en restreignant le champ d'application, des solutions commencent à émerger.

Nous avons vu que les travaux menés jusqu'à présent touchent un large panel de langues. En ne s'intéressant qu'à la reconnaissance de caractères, la diversité des types de formes rencontrées au travers des langues impose des approches différentes.

1.2.1 Les chiffres manuscrits

La reconnaissance de chiffres arabes manuscrits est considérée comme la plus simple à réaliser. Le vocabulaire est très restreint (10 chiffres), l'écriture est relativement contrainte (des cadres sont souvent utilisés à l'exemple des codes postaux, des dates de naissance...) et les formes sont assez simples. Les résultats présentés par LECUN & AL sur la base MNIST ([LBBH98], figure 1.8) montrent qu'une

FIGURE 1.8 – Exemple de chiffres normalisés extraits de la base MNIST ([LBBH98]).

description des images de chiffre normalisées utilisant le vecteur des pixels binaires suffit à une reconnaissance supérieure à 95%.

1.2.2 Les écritures alphabétiques

Un alphabet (de alpha et bêta, les deux premières lettres de l'alphabet grec) est un ensemble de symboles utilisé pour représenter plus ou moins précisément les phonèmes d'une langue.

Chacun de ces symboles, ou graphèmes, est appelé une lettre ; chaque lettre, en théorie, devrait noter un phonème. Certaines lettres peuvent recevoir un ou plusieurs accents (ou *diacritiques*) afin d'étendre le stock de graphèmes si celui-ci est insuffisant pour noter les sons de la langue ou permettre d'éviter les ambiguïtés. De la même manière, un alphabet peut être étendu par l'utilisation d'un assemblage de deux graphèmes (ou *digramme*) ou encore de lettres supplémentaires.

1.2.2.1 Les écritures latines et grecques

L'alphabet latin était initialement utilisé pour écrire le latin, la langue parlée par les habitants de Rome et du Latium. Il est dérivé de l'alphabet étrusque, lui-même variante d'un alphabet grec duquel dérive l'alphabet grec dit classique (celui utilisé dans les éditions actuelles). En conséquence de quoi, les écritures latines et grecques sont relativement similaires quant à leur complexité ainsi qu'aux traitements nécéssaires par les RAED dont ils sont l'objet.

L'alphabet latin, comme la majorité de ceux issus de l'alphabet grec, est *bicaméral* : deux graphies sont utilisées pour chaque graphème (ou lettre), l'une dite minuscule, l'autre capitale. Dans la majorité des cas, chaque lettre possède les deux variantes. Il existe cependant quelques exceptions, comme la lettre 'ß' (eszett ; utilisée en allemand et autrefois dans d'autres langues, dont le français), qui, en capitale, est remplacée par SS. Le français compte 26 lettres ainsi qu'un digramme lié ('œ') et une lettre supplémentaire ('ç'), soit un total de 28 caractères et 56 symboles (avec les capitales).

L'alphabet grec provient, quant à lui, de l'alphabet phénicien. Il compte actuellement vingt-quatre lettres et une variante contextuelle du sigma ('σ' en début de mot et 'ς' au milieu d'un mot). D'anciens caractères, maintenant inusités, peuvent également se rencontrer. Une lettre ancienne abandonnée après l'adoption du modèle ionien (en -403) se retrouve souvent sur les transcriptions épigraphiques : le digamma (minuscule 'ϝ' et majuscule 'F'). Quatre chiffres sont également utilisés dans les manuscrits médiévaux : le sampi ('ϡ' qui vaut 900), le koppa ('ϟ' qui vaut 90) ,une variante du digamma ('ϛ' qui vaut 6) et le stigma ('ϛ' qui vaut également 6). Le tout donne un total de 56 symboles comme le français.

Ces écritures sont à vocabulaire relativement limité. Cependant la variabilité des écritures au cours du temps et entre scripteurs, ainsi que leur aspect non-contraint rend leur reconnaissance relativement complexe. Du fait de la relative simplicité des formes, les decriptions sont souvent de type statistique ou simplement binaire (vecteurs de pixels). En revanche les techniques de classification sont plus variables et englobent la quasi totalité des approches décrites en 1.1.1 page 5, utilisées seules ou en coopération ([Bun03]).

1.2.2.2 L'écriture arabe

L'alphabet arabe (figure 1.10 page suivante) comprend vingt-neuf lettres fondamentales (vingt-huit si l'on exclut la hamza, qui se comporte, soit comme une lettre à part entière, soit comme un diacritique). Le sens d'écriture va de droite à gauche. Il n'y a pas de différence entre les lettres manuscrites et les lettres imprimées ; les notions de lettre capitale et lettre minuscule n'existent pas. En revanche, la plupart des lettres s'attachent entre elles, même en imprimerie, et leur graphie diffère selon qu'elles sont précédées et/ou suivies d'autres lettres ou qu'elles sont isolées (variantes contextuelles). Certaines lettres, cependant, ne s'attachent jamais à la lettre suivante : de fait, un mot unique peut être entrecoupé d'un ou plusieurs espaces, lesquels sont aussi utilisés pour séparer les mots. La longueur de cet espace inter-mot est généralement supérieure à l'espace intra-mot entre caractères non attachés.

L'alphabet arabe est un abjad, le lecteur doit connaître la structure de la langue pour restituer les voyelles. Cela se traduit par le fait que toutes ces lettres, à l'exception du Alef, sont des consonnes. Le Waw et le Ya sont, elles, des demi-voyelles, dans la mesure où elles représentent à la fois une consonne et une voyelle : le Waw se prononce 'w' ou 'ou' alors que le Ya se prononce 'y' ou 'i'. Ainsi, la racine trilitère 'KTB' (figure 1.9 page suivante) peut, selon le vocalisme, être lue 'kutub' (*les livres*), 'kataba' (*il écrivit*) ou encore 'kutiba' (*qui a été écrit*).

Cet alphabet remonte à l'araméen lui-même descendant du phénicien (alphabet qui, entre autres, donne naissance à l'alphabet hébreu, à l'alphabet grec et, partant, au cyrillique, aux lettres latines, etc.). La première attestation d'un texte en alphabet arabe remonte à 512. C'est au VIIe siècle que les diacritiques furent rajoutées sur ou sous certaines lettres afin de les différencier, le modèle araméen ayant moins de phonèmes que l'arabe.

Tel qu'il est écrit couramment, l'alphabet arabe n'utilise pour ainsi dire pas de diacritiques, outre le point souscrit ou suscrit obligatoire pour distinguer des lettres ambiguës. Cependant, afin de faciliter la lecture, et ce dans un cadre didactique ou religieux, de nombreux signes auxiliaires sont rajoutés pour rendre la lecture du texte moins ambigüe. Sans cela il serait difficile à un lecteur débutant de lire à voix haute un texte sans une bonne connaissance de la langue.

L'écriture arabe est un cas très intéressant en RAED de par son vocabulaire limité et fortement non contraint (les lettres sont cursives et attachées). De plus les diacritiques rendent la reconnaissance encore plus délicate. Les techniques utilisées sont celles de la reconnaissance de mots du paragraphe 1.1.2 page 6 même si le problème de la segmentation n'est pas encore totalement résolu. Enfin les systèmes

FIGURE 1.9 – Racine trilère ’KTB’ en arabe.

Caractère (forme finale)	Equivalent phonétique	Nom	Caractère (forme finale)	Equivalent phonétique	Nom
ا	a	Alef	ك	k	Kaf
ض	d	Dad	ذ	d	Thal
ب	b	Ba'	ل	l	Lam
ط	t	Tah	ر	r	Ra
ت	t	Ta'	م	m	Mim
ظ	z	Zah	ز	z	Zayn
ث	th	Tha'	ن	n	Nun
ع	'	'Ayn	س	s	Sin
ج	j	Jim	ه	h	Ha
غ	gh	Ghayn	ش	sh	Shin
ح	h	Hha'	و	w	Waw
ف	f	Fa	ص	s	Sad
خ	kh	Kha'	ى	y	Ya
ق	q	Qaf	ء	,	Hamzah
د	d	Dal			

FIGURE 1.10 – Alphabet Arabe.

actuels ne traitent que les écrits sans vocalisation donc sans diacritique ([Ami98]).

1.2.3 Les écritures idéographiques

Plus de la moitié de la population mondiale lettrée d’aujourd’hui utilise une écriture idéographique. Du fait du formidable essort économique des pays d’Asie et de la Chine en particulier, la problématique de la reconnaissance de ce type d’écriture est fortement d’actualité dans la communauté scientifique.

Par définition, un idéographe représente une idée par un signe qui en figure l’objet. Cependant, toutes les écritures idéographiques ont évolué et comprennent maintenant des caractères phonétiques. L’utilisation du terme idéogramme est donc abusive mais employée par commodité.

1.2.3.1 Le chinois

Le système d’écriture chinoise à base de sinogrammes, appelés *Hanzi*, s’est répandu dans toute l’Asie. Il est utilisé également par les Japonnais, pour les caractères courants appelés *Kanji* ainsi que par les Coréens, en tant que caractères optionnels appelés *Hanja*. Les caractères Hanzi sont issus d’une évolution d’idéogrammes de plus de 4 000 ans. Le *Kangxi* publié en 1716 se veut un classique des dictionnaires chinois. Il contient près de 50 000 caractères compilés en 42 fascicules. La plupart de ces

FIGURE 1.11 – Table des radicaux chinois (*http ://www.lechinois.com/*) : (a) les 214 radicaux traditionnels, (b) les variantes, (c) les simplifications.

caractères représentent des variantes rares qui se sont accumulées au cours des siècles. Ces caractères sont regroupés sous 214 radicaux (figure 1.11, (a)) qu'utilisent toujours plusieurs dictionnaires pour classer les caractères selon leur forme traditionnelle avec quelques variantes (figure 1.11, (b)).

La première version de la liste des caractères simplifiés est apparue en 1956 (les radicaux simplifiés de cette liste sont présentés à la figure 1.11, (c)). Plusieurs ajouts et modifications ont été apportés par la suite à cette liste. Elle dénombre environ 6 500 caractères simplifiés. Actuellement les caractères simplifiés sont utilisés en Chine et à Singapour. Alors que Taiwan, Hongkong et la plupart des communautés chinoises d'outre-mer continuent d'utiliser les caractères traditionnels.

Les fontes chinoises utilisées sur les ordinateurs pour les traitements de texte comptent environ 6 500 caractères pour la forme simplifiée et 13 500 pour la forme traditionnelle. Pour illustrer la complexité des caractères chinois, KANAI & LIU ([KL97]) ont réalisé quelques statistiques sur le texte d'instructions nécessaires pour remplir un formulaire de déclaration de douane américaine :

Before arriving in the United States of America, each traveler or head of family is required to fill out a Customs Declaration Form.
Most of the questions can be answered with "yes" or "no". The reverse side of this form must be signed and dated.
Please print legibly, using black or blue ink. Entries must be in ENGLISH and in ALL CAPITAL LETTERS.
The Customs Declaration form will be distributed during the flight.

La transcription en caractères Hanzi de ce texte est présentée à la figure 1.12.

在抵达美国前，每位旅客都需填写一张海
关申请表，但是每个家庭只需填写一张。
报关表上多数问题只需用“不是”或“是”
即可回答清楚，请务必在背面签名并填上
申请日期。
请你用蓝色或黑色墨水，以英文大写的正
楷仔细填写。报关表会由空服员在飞机降
落前发给每一位乘客。

FIGURE 1.12 – Instructions pour remplir un formulaire de déclaration de douane américaine, caractères Hanzi.

KANAI & LIU comptent 346 symboles au total (sans compter les espaces) dans le texte anglais contre 112 dans la transcription en caractères Hanzi. En revanche ils ne dénombrent que 44 symboles uniques dans le texte anglais contre 93 dans la transcription en caractères Hanzi.

Malgré la classification en radicaux, il n'y a pas d'autre solution pour lire le chinois que de connaître le maximum de caractères. Certains caractères sont composés et peuvent se comprendre par déduction mais malheureusement ce n'est pas souvent le cas. En revanche, du point de vue de leur construction, tous les caractères chinois sont calligraphiés à partir de huit traits fondamentaux (figure 1.13 page suivante) :

1. le « point » qui est un segment très court isolé ;
2. le trait horizontal ;
3. le trait vertical ;
4. le relevé qui est un trait montant de gauche à droite ;
5. le jeté qui est un trait descendant de droite à gauche ;
6. le jeté court qui est un trait descendant de gauche à droite ;
7. le trait brisé ;
8. le crochet.

En résumé, le chinois est une écriture à très large vocabulaire, qui ne peut pas être traitée caractère par caractère. Néanmoins, c'est une écriture contrainte et dont les traits fondamentaux sont bien définis. Les techniques de reconnaissance de mots ne peuvent pas s'appliquer facilement à ce type d'écriture puisque la structure du caractère est bidimensionnelle. Les premières études sur ce type d'écriture se sont

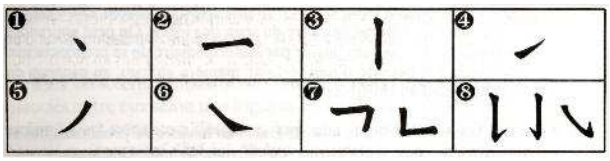


FIGURE 1.13 – Les huit traits fondamentaux du chinois.

orientées naturellement vers des descriptions structurelles, de type graphe, ou syntaxiques. Cependant les difficultés à extraire les traits et la sensibilité aux bruits ont conduit les recherches vers des systèmes hybrides alliant une description structurelle à une classification statistique [Din97].

1.2.3.2 Les hieroglyphes égyptiens

L'apparition des premiers hiéroglyphes remonte à 3 500 av. J.-C. La dernière inscription hiéroglyphique connue est datée du 24 août 394, et se trouve au temple de Philae. D'abord exclusivement figurative, l'écriture hiéroglyphique s'enrichit avec le temps de signes en rendant la lecture symbolique. Alors qu'il existe environ 700 hiéroglyphes à l'époque archaïque, pas moins de 5000 à l'époque la plus tardive sont dénombrés (époque gréco-romaine).

CHAMPOLLION [Cha22] indiquait, dans sa « Lettre à M. Dacier », à propos du système hiéroglyphique : « C'est un système complexe, une écriture tout à la fois figurative, symbolique et phonétique, dans un même texte, une même phrase, je dirais presque dans un même mot. ». Chaque signe peut être à la fois un *logogramme* représentant l'objet qu'il décrit, un *signe consonnant* associé à une valeur phonétique ou encore un *signe déterminatif* qui apporte une information sémantique [CM98]. Quelques exemples sont donnés dans la figure 1.14 page suivante.

L'écriture peut se faire séquentiellement de droite à gauche ou de gauche à droite (l'orientation des caractères donne l'indication) ou encore par bloc de hiéroglyphes qui se lit alors de haut en bas. De plus, seules les consonnes sont écrites, comme en arabe. La figure 1.15 page suivante donne un exemple simple de la transcription du mot « jour ». Remarquons la présence du caractère sémantique « soleil » qui n'apporte pas d'information phonétique, il n'est là que pour accentuer la signification du mot. Comme la langue égyptienne a été en partie perdue et que les voyelles ne sont pas écrites, la convention veut que les consonnes soient séparées par un « i » (« e » en anglais) pour la prononciation. Ainsi « jour » se prononce « hiriwi ».

Aujourd'hui les hiéroglyphes s'utilisent essentiellement dans les transcriptions manuscrites des égyptologues. Toutes les archives accumulées par les historiens, parfois numérisées, ne sont pas, à notre connaissance, exploitées pour la RAED. Il n'existe pas non plus de travaux de recherche sur la reconnaissance de ces écritures en dehors de quelques applications illustrant des techniques de reconnaissance de forme. En revanche, il existe depuis 1988 un manuel pour le codage informatique des hiéroglyphes [BGH$^+$88] qui est à la base de la totalité des systèmes professionnels utilisés en égyptologie. EVERSON [Eve99] a proposé, en 1999, un codage ISO 10646 des hiéroglyphes reposant sur ce manuel.

L'écriture hiéroglyphique se rapproche du chinois en ce qu'elle est relativement contrainte et à large vocabulaire. Cependant, ne reposant pas sur un nombre limité de graphèmes, elle nécessite une décomposition structurelle plus complexe.

	Logogramme	Consonnance	Déterminant
	*Une chouette*	*m*	N.D.
	*Une bouche*	*r*	N.D.
	*Une maison*	*pr*	*lieu, construction*
	*Les jambes*	N.D.	*marcher, avancer, venir, s'arrêter*
	*Plan d'édifice*	*h*	N.D.
	*Poussin de caille*	*w*	N.D.
	*Le soleil*	N.D.	*soleil, lumière, jour, divisions du temps*

FIGURE 1.14 – Exemple de hieroglyphes et leur signification en tant que logogramme, signe consonnant et/ou signe déterminatif (N.D. indique une valeur Non Définie).

Caractère	Valeur phonétique	Indication sémantique	Signification
	*hrw*	*soleil, lumière, jour*	*Le jour*

FIGURE 1.15 – Transcription du mot jour en hiéroglyphes.

1.3 Problématique scientifique

1.3.1 Cadre et orientations

Les travaux présentés dans ce mémoire se sont déroulés dans le cadre du projet COROC[6] porté par l'entreprise RC-SOFT[7] et qui bénéficie du label EUREKA depuis le 27 juin 2002. RC-SOFT est une PME française, soutenue par l'ANVAR, effectuant des travaux de recherche et de développement informatique, qui développe un logiciel « auto-apprenant » de reconnaissance de caractères manuscrits cursifs (documents anciens notamment). Ce système OCR veut s'affranchir des bibliothèques de langues et de la forme des caractères. Il doit reconnaître après une phase d'apprentissage automatique la plupart des caractères.

Ces travaux ont comme objectif d'explorer les méthodes de reconnaissance générique. Ils se sont donc naturellement orientés vers les approches structurelles qui nous ont semblé être les plus intéres-

[6] Cognitive Optical Recognition Old Characters
[7] http ://www.rcsoft.fr

santes dans cette optique. Nous avons focalisé notre attention sur l'étape de description et de reconnaissance des caractères. Le pré-traitement et la segmentation ne seront pas abordés (ou très peu) dans ce mémoire et les résultats présentés le sont sur des bases déjà segmentées et non traitées.

1.3.2 Bases de validation

Nous présentons ici rapidement les bases qui ont servi aux tests présentés dans les trois paragraphes de résultats en fin de chaque partie. Une présentation plus détaillée est donnée en annexe A page 155. Les bases sont de trois types, trois bases de caractères, une base de mots arabes et une base de hiéroglyphes. Toutes sont manuscrites multi-scripteurs et considérées comme bien segmentées. Les rares prétraitements effectués sont décrits dans l'annexe.

Les trois premières bases sont des bases de caractères. La première est constituée de caractères grecs manuscrits écrites par quatre scripteurs dans le cadre de COROC. Elle sera désignée par le nom « GrCor » dans le reste du manuscrit. Elle est relativement propre et constitue la base la moins complexe. Elle nous a servi de base de référence pour toutes les méthodes testées et pour le choix des paramètres. La deuxième est une base de caractères grecs extraits de quatre pages de manuscrits anciens de scripteurs et d'époques différents. Elle est notée « GrAnc » dans le mémoire. C'est une petite base qui nous a servi également pour choisir certains paramètres de test laborieux à optimiser sur de grandes bases. Enfin la troisième base est la base MNIST[8] de chiffres manuscrits et notée « MNIST ». Elle présente l'avantage d'être une base de référence dans la communauté scientifique ([LBBH98]).

La base de mots arabes est la base de noms de villes tunisiennes de l'IFN/ENIT. Elle a été utilisée comme base de compétition à ICDAR 2005 (International Conference on Document Analysis and Recognition) et les résultats sont présentés dans l'article de MÄRGNER ET COLL. ([MPEA05]). Nous avons utilisé cette base pour tester notre système de reconnaissance à base de graphes d'attributs hiérarchiques flous présenté dans la troisième partie du mémoire.

Enfin la base de hiéroglyphes notée « HrMan » est une base non indexée. C'est une série de caractères issue d'une page manuscrite écrite par un égyptologue. Une autre page manuscrite écrite par un autre égyptologue sert de base de test. Elle n'a été testée que dans la troisième partie du mémoire.

1.3.3 Contributions

Dans cet exposé nous allons proposer deux chaînes de reconnaissance à base de graphes. A travers ces deux chaînes, notre ambition n'est pas de présenter un système entièrement fonctionnel mais de regarder en quoi les graphes peuvent aider à la reconnaissance non-contrainte de caractères manuscrits.

Le premier est un système probabiliste qui nécessite un apprentissage et est donc relativement dédié. Il a été testé en système multi-scripteurs et n'est, en l'état, applicable qu'aux écritures à vocabulaire réduit et dédié aux bases de caractères. Nous présentons les apports d'un tel système par rapport aux approches statistiques classiques. Ce système a fait l'objet de deux publications scientifiques, la première au huitième Colloque International Francophone sur l'Ecrit et le Document en juin 2004 ([ARFM+04], session orale), la seconde au cinquième « IAPR International Workshop Graph-Based Representations in Pattern Recognition » en avril 2005 ([ARFM+05], session poster).

Le second est un système flou générique qui n'utilise pas d'apprentissage. C'est un système basé modèles qui s'applique à tout type d'écriture et est omni-scripteurs. Il a fait l'objet de deux publications

[8] http ://yann.lecun.com/exdb/mnist/

scientifiques, la première à la huitième « International Conference on Document Analysis and Recognition » en août 2005 ([ARB05b], session orale) et la seconde à la onzième « International Conference on Computer Analysis of Images and Patterns » en septembre 2005 ([ARB05a], session poster).

Ces travaux s'inscrivent dans un travail d'équipe sur la modélisation des connaissances images au sein du Laboratoire Signal Image et Communication de Poitiers. Ils ont à ce titre fait l'objet de plusieurs communications orales et écrites.

1.4 Présentation du mémoire

Ce mémoire est organisé en trois parties. La première partie porte sur les approches dites statistiques et plus particulièrement les approches probabilistes. La seconde partie s'intéresse aux méthodes structurelles en reconnaissance de forme et introduit dans ce contexte notre système de reconnaissance graphique probabiliste. La troisième partie est dédiée au système de reconnaissance flou.

La première partie de ce mémoire est composée de trois chapitres. Le premier chapitre se propose de redéfinir les différentes descriptions statistiques utilisées en reconnaissance de formes ainsi que les métriques qui leur sont associées. Le second chapitre introduit les différentes approches probabilistes classiques de reconnaissance utilisant ces descripteurs. Nous nous focalisons sur les approches paramétriques et non-paramétriques et rappelons la stratégie de décision bayésienne avec rejets. Cette stratégie nous permet de donner quelques résultats commentés dans le troisième et dernier chapitre de cette partie.

La seconde partie est relativement symétrique à la première. Le premier chapitre commence par une étude des différents algorithmes de squelettisation de formes qui sont à la base des descriptions structurelles des caractères. Il expose ensuite brièvement les techniques d'extraction de points singuliers utilisées dans notre système et finit par un rappel des différentes approches de descriptions structurelles en mettant en relief l'intérêt du choix des graphes dans notre problématique. Le deuxième chapitre s'attache à décrire notre système de reconnaissance probabiliste à base de graphes en justifiant les choix scientifiques opérés. Enfin la troisième partie donne les résultats de reconnaissance du système seul et propose un système coopérant avec les approches statistiques décrites dans la première partie. Les résultats du système seul et en coopération sont analysés et discutés.

La troisième partie est constituée de deux chapitres. Le premier expose notre second système de reconnaissance basé sur une description des formes à l'aide du nouveau concept de graphe d'attributs hiérarchiques flous. Ce concept est introduit dans son cadre scientifique avec son formalisme. Nous expliquons ensuite les choix effectués pour la description et la reconnaissance avant de donner un schéma général du système. Le deuxième chapitre présente les résultats obtenus avec ce système sur les différentes bases de test.

Enfin nous finirons cet exposé par un chapitre de conclusion en essayant de mettre en évidence les avantages et les limitations des deux systèmes présentés. Nous terminerons en donnant quelques perspectives à ces travaux.

Première partie

Les approches statistiques

CHAPITRE 2

DESCRIPTION NUMÉRIQUE D'UNE FORME ET CALCUL DE DISTANCE

Sommaire

La reconnaissance d'une forme est basée sur une description de celle-ci. La description peut ainsi être vue comme une modélisation mathématique de l'objet permettant un calcul de distance entre deux ou plusieurs modèles. La notion de distance est à prendre ici au sens large comme une mesure de ressemblance.

Il existe un très grand nombre de descriptions possibles pour une forme suivant l'information à modéliser ([TJT96]) et les invariances à exprimer. LONCARIC ([Lon98]) distingue trois classifications possibles pour les descripteurs de forme :

- La première, la plus courante, vient de PAVLIDIS ([Pav78]) qui considère d'une part les descripteurs basés sur la frontière (*descripteurs externes*) et d'autre part ceux basés sur la forme elle-même (*descripteurs globaux ou internes*).
- La seconde consiste à différencier les descripteurs numériques (scalaires ou vectoriels), des descripteurs qui ne le sont pas, également désignés comme techniques appartenant au domaine spatial.
- Enfin la troisième classe les descripteurs selon qu'ils préservent entièrement ou seulement partiellement l'information contenue dans la forme. Autrement dit peut-on reconstruire la forme initiale à partir des descripteurs ? La notion d'inversibilité est dépendante des normalisations effectuées.

Nous exposons, ici, différentes approches numériques utilisées en description de formes ainsi que les distances qui permettent d'entrer dans un processus de reconnaissance. Les approches structurelles seront détaillées dans la partie II page 61 de ce mémoire.

2.1 Distances et similarités, quelques rappels

Distance et similarité sont des notions de ressemblance entre individus. Ce sont des applications particulières du carré de l'espace de représentation des individus dans $\mathbb{R}^+$. Supposons que trois formes a, b et c soient décrites dans un espace de représentation E. Une distance d est une application $E \times E \rightarrow \mathbb{R}^+$ telle que :

1. $d(a,b) = d(b,a)$, d est symétrique,
2. $d(a,b) = 0$ si et seulement si $a = b$, d remplit la propriété d'identité,
3. $d(a,b) + d(b,c) \geq d(a,c)$, d remplit l'inégalité triangulaire.

Le couple (E,d) est alors appelé espace métrique. Il peut également être défini des notions de ressemblance relevant d'une axiomatique moins stricte, il s'agit alors d'une *pseudo-distance*. Une *quasi-métrique*, par exemple, n'est pas symétrique ; une *semi-métrique* ne satisfait pas l'inégalité triangulaire et si les deux conditions ne sont pas remplies, il s'agit d'une *pré-métrique*.

La notion de distance correspond à ce qui sépare, celle de similarité est relative à ce qui rapproche les individus. Mathématiquement toute distance correspond à plusieurs indices de similarité et réciproquement, ces deux notions sont symétriques. Ainsi une similarité s est une application $E \times E \rightarrow [0, s_{max}]$ telle que :

1. $s(a,b) = s(b,a)$, s est symétrique,
2. $s(a,b) = s_{max}$ si et seulement si $a = b$.

Si d est une distance sur $E \times E$ alors une similarité est définie par $s_1(a,b) = s_{max}/(d(a,b)+1)$ ou bien par $s_2(a,b) = s_{max} - \dfrac{d(a,b)}{\max\limits_{i,j \in E \times E} d(i,j)}$ si la distance est bornée.

2.2 Les descripteurs numériques

Les descriptions numériques résument l'information contenue dans la forme entière par un vecteur d'attributs. Il existe une multitude de descripteurs dans la littérature et la liste de ceux présentés ici ne se veut pas exhaustive.

2.2.1 Descripteurs globaux

L'image I d'une forme peut être décrite par une application $f : (x,y) \rightarrow \{0,1\}$ où $(x,y) \in I$ sont les coordonnées des pixels de l'image. Si le pixel (x,y) est en dehors de la forme $f(x,y) = 0$ et $f(x,y) = 1$ s'il appartient à la forme. l'ensemble des points de la forme est noté $F = \{(x,y) \in I | f(x,y) = 1\}$ et $I(n \times m)$ est le support de l'image avec m le nombre de lignes et n le nombre de colonnes.

2.2.1.1 Template Matching

Ces techniques font partie des méthodes d'extraction de caractéristiques relatives à un modèle. Soit f_0 décrivant le modèle d'une forme et f l'application décrivant une forme inconnue dont les caractéristiques relatives sont recherchées. Si les images sont de même support, ces méthodes fournissent directement une mesure de distance ou de similarité qui peut être utilisée telle quelle dans une phase de reconnaissance.

Images de même support Si les deux images présentent exactement le même support I, n_{ij} est défini comme le nombre de pixels de coordonnées (x, y) de I tels que $f(x, y) = i$ et $f_0(x, y) = j$ avec $i, j \in \{0, 1\}$. Il en découle la relation :

$$n \times m = n_{11} + n_{00} + n_{10} + n_{01} \tag{2.1}$$

La métrique la plus couramment utilisée pour comparer deux images pixel à pixel est la *distance de Manhattan ou city block encore appelée distance de Hamming* (équation 2.2). Le principal inconvénient d'une telle approche est son extrême sensibilité aux bruits ainsi qu'à toute transformation affine de la forme.

$$D_{Manhattan}(F, F_0) = \sum_{(x,y)\in I} |f_0(x, y) - f(x, y)| = n_{10} + n_{01} \tag{2.2}$$

Le principe de la *distance de Hausdorff* est une alternative intéressante pour diminuer la sensibilité aux bruits ([Ruc96], [SKP99]) :

$$D_H(F, F_0) = \max(h(F_0, F), h(F, F_0)) \tag{2.3}$$

avec :

$$h(F, F_0) = \max_{a\in F}(\min_{b\in F_0} \|a, b\|) \tag{2.4}$$

où $\|a, b\|$ est une métrique définie entre les deux points (distance euclidienne par exemple). La distance de Hausdorff reste sensible aux points isolés puisqu'elle correspond au maximum de toutes les distances mesurables entre chaque point de F et le point le plus proche de F_0. Afin de diminuer cette sensibilité, RUCKLIDGE ([Ruc96]) a défini la *distance de Hausdorff partielle*. Il range les points de F en fonction de leur distance minimale aux points de F_0 et substitue le choix du maximum de l'équation 2.4 par le $k^{ième}$ maximum dépendant de l'application :

$$D_H^k(F, F_0) = k^{ième}_{a\in F}(\min_{b\in F_0} \|a, b\|) \tag{2.5}$$

DUBUISSON & JAIN ([DJ94]), quant à eux, ont établi une *distance de Hausdorff modifiée* en remplaçant le choix du maximum dans l'équation 2.4 par une moyenne sur les points de F :

$$D_H^{mod}(F, F_0) = \frac{1}{n \times m} \sum_{a\in F}(\min_{b\in F_0} \|a, b\|) \tag{2.6}$$

TUBBS ([Tub89]) a évalué plusieurs mesures de similarité entre images binaires décrites par des applications f et f_0 de même support I. Il a montré que la mesure de *similarité de Jaccard ou de Tanimoto* (s_J, équation 2.7) et la mesure de *similarité de Yule* (s_Y, équation 2.8) donnaient les meilleurs résultats avec des images non bruitées.

$$s_J(F, F_0) = \frac{n_{11}}{n_{11} + n_{10} + n_{01}} \tag{2.7}$$

$$s_Y(F, F_0) = \frac{n_{11}n_{00} - n_{10}n_{01}}{n_{11}n_{00} + n_{10}n_{01}} \tag{2.8}$$

Ces deux similarités sont à valeurs dans $[0, 1]$ pour la première et [-1,1] pour la seconde. FLIGNER & AL ([FVBB01]) proposent également une mesure de *similarité de Jaccard modifiée* afin de prendre en compte l'appariement des pixels blancs :

$$s_J^{mod}(F, F_0) = \alpha \times s_J(F, F_0) + (1 - \alpha) \times (s_J(F, F_0))^C \tag{2.9}$$

où $\alpha \in [0, 1]$ et

$$(s_J(F, F_0))^C = \frac{n_{00}}{n_{00} + n_{10} + n_{01}} \tag{2.10}$$

Si α vaut 1, seuls les pixels noirs sont pris en compte et s'il vaut 0 la similarité repose uniquement sur les pixels blancs. Enfin, parmi tant d'autres méthodes, LIOLIOS & AL ([LKFK02]) proposent de calculer un coût de transformation de F en F_0 en décomposant cette opération en trois opérations primitives :

– la substitution d'un pixel de F par un pixel de F_0 ayant les mêmes coordonnées,
– le déplacement d'un pixel de F vers un pixel voisin de F_0,
– la suppression de pixels de F ou de F_0.

Les résultats présentés, obtenus sur la base NIST 19 ([Gro95]), montrent des performances intéressantes par rapport aux descripteurs de Karhunen-Loève (c.f. paragraphe 2.2.1.2).

Images de supports différents Si maintenant les supports sont différents mais que les formes qu'ils contiennent sont de même taille, par exemple $I_0(n_0 \times m_0)$ est plus petit que $I(n \times m)$ ($n_0 < n$ et $m_0 < m$), alors une *matrice de corrélation croisée* définie sur $(n - n_0, m - m_0)$ peut être utilisée :

$$M[i, j] = \frac{\sum_{(k,l)\in I_0} f_0(k, l) f(k + i, l + j)}{\sqrt{\sum_{(k,l)\in I_0} f(k + i, l + j)^2}} \tag{2.11}$$

Les valeurs de M les plus élevées indiquent les positions de I où f est similaire à f_0. La description ainsi obtenue n'est pas vectorielle mais matricielle, elle est essentiellement utilisée pour retrouver un motif dans une scène.

2.2.1.2 La transformée de Karhunen-Loève

Les techniques de Template Matching décrivent une image binaire relativement à un modèle. La transformée de Karhunen-Loève ([Jai89]), quant à elle, utilise un ensemble d'images binaires pour définir une base d'un espace vectoriel dans laquelle chaque caractère est décrit par un vecteur de coordonnées.

Considérons un ensemble de P images binaires de support $I = (N \times N)$ et caractérisées par les applications $f^{(p)}$, $p = 1, \ldots, P$. A chaque image est associée une matrice $U^{(p)} = \{u_{ij}^{(p)}\}_{i,j=1,\ldots,N}$ (équation 2.12) et un vecteur $\mathbf{u^{(p)}}$ (équation 2.13) composé des colonnes de $U^{(p)}$ mises bout à bout.

$$u_{ij}^{(p)} = \begin{cases} +1 \; si \; f^{(p)}(i, j) = 1 \\ -1 \; si \; f^{(p)}(i, j) = 0 \end{cases} \tag{2.12}$$

$$\mathbf{u^{(p)}} = \{u_{11}^{(p)}, u_{21}^{(p)}, \ldots, u_{N1}^{(p)}, u_{12}^{(p)}, \ldots, u_{N2}^{(p)}, \ldots, u_{NN}^{(p)}\} \tag{2.13}$$

Chaque image est représentée par un vecteur $\mathbf{u^{(p)}}$ et le vecteur moyen de ces P vecteurs s'exprime simplement de la manière suivante :

$$\bar{\mathbf{u}} = \frac{1}{P} \sum_{p=1}^{P} \mathbf{u^{(p)}} \tag{2.14}$$

La matrice de covariance de l'ensemble des vecteurs $\mathbf{u}^{(\mathbf{p})}$ centrés s'exprime :

$$R = E\left\{(\mathbf{u} - \bar{\mathbf{u}})(\mathbf{u} - \bar{\mathbf{u}})^T\right\} \tag{2.15}$$

soit :

$$R = \frac{1}{P}\sum_{p=1}^{P}\left(\mathbf{u}^{(\mathbf{p})}\mathbf{u}^{(\mathbf{p})}\right)^T - (\bar{\mathbf{u}}\bar{\mathbf{u}})^T \tag{2.16}$$

R possède N^2 vecteurs propres colonnes, rassemblés dans la matrice Φ, associés à M valeurs propres non nulles λ_i éléments diagonaux de la matrice Λ :

$$R\Phi = \Phi\Lambda \tag{2.17}$$

Λ et Φ sont ordonnées suivant les valeurs décroissantes de λ_i. Les vecteurs propres représentent les axes de maximum de variation dans l'espace de dimension N^2. Tout vecteur $\mathbf{u}^{(\mathbf{p})}$ peut s'exprimer dans le nouveau repère par :

$$\mathbf{u}^{(\mathbf{p})} = \Phi\mathbf{v}^{(\mathbf{p})} + \bar{\mathbf{u}} \tag{2.18}$$

où $\mathbf{v}^{(\mathbf{p})}$ représente la transormée de Karhunen-Loève de $\mathbf{u}^{(\mathbf{p})}$. En réduisant Φ aux k vecteurs propres correspondant aux k plus grandes valeurs propres, $\hat{\mathbf{u}}^{(\mathbf{p})}$ est une estimation de $\mathbf{u}^{(\mathbf{p})}$ avec comme erreur de reconstruction e :

$$\hat{\mathbf{u}}^{(\mathbf{p})} = \Phi_k\mathbf{v}^{(\mathbf{p})} + \bar{\mathbf{u}} \tag{2.19}$$

$$e = \sum_{i=1}^{N^2}\lambda_i - \sum_{i=1}^{k}\lambda_i = \sum_{i=k+1}^{N^2}\lambda_i \tag{2.20}$$

2.2.1.3 Les descripteurs simples

Une forme binaire peut intuitivement se décomposer en formes primitives avec des caractéristiques géométriques associées ([PI97]). Voici quatre caractéristiques couramment utilisées car invariantes par translation, rotation et changement d'échelle.

Définition 1 (Rectangularité) *Le rapport de la surface d'une forme avec celle de sa boîte englobante*[1] *est appelé* rectangularité *(figure 2.1 page suivante(a)) et se définit suivant :*

$$Rectangularité = \frac{\mathcal{A}_{forme}}{\mathcal{A}_{boîte\ englobante}} \tag{2.21}$$

Le calcul du rectangle englobant minimum est une opération un peu délicate pour laquelle ROSIN propose une solution intéressante ([Ros99]).

Définition 2 (Convexité) *Le rapport du périmètre d'une forme avec celui de son enveloppe convexe ([BIK+02]) est appelé* convexité *(figure 2.1 page suivante(b)) et se définit par :*

$$Convexité = \frac{\mathcal{P}_{forme}}{\mathcal{P}_{enveloppe\ convexe}} \tag{2.22}$$

Définition 3 (Compacité) *Le rapport du carré du périmètre d'une forme à sa surface est appelé* compacité *(figure 2.1 page suivante(c)) et se définit par :*

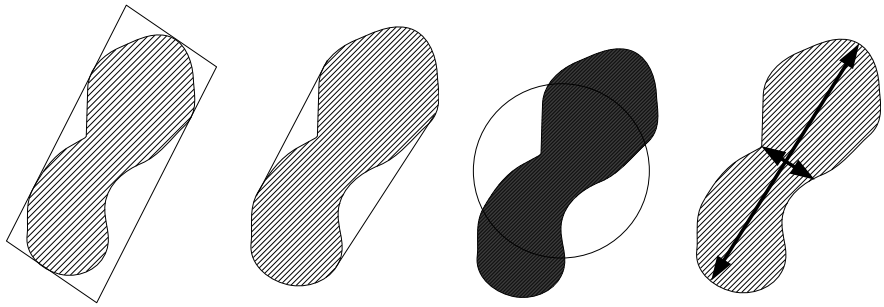
$$Compacité = \frac{\mathcal{P}^2_{forme}}{\mathcal{A}_{forme}} \tag{2.23}$$

[1] rectangle de plus petite aire contenant la forme

Définition 4 (Excentricité) *Le rapport entre la longueur de l'axe principal et celle de l'axe secondaire d'une forme est appelé* excentricité *ou* ellipticité *(figure 2.1(d)) et se définit par :*

$$Excentricité = \frac{\mathcal{L}_{grand\ axe}}{\mathcal{L}_{petit\ axe}} \tag{2.24}$$

Il existe plusieurs façons de calculer l'excentricité. PEURA & AL ([PI97]) proposent naturellement d'utiliser les axes d'inertie de la décomposition en composantes principales du nuage de points de la forme. Un calcul plus direct à partir des moments centrés d'ordre 2 est exposé par ROSIN ([Ros03]).



(a) (b) (c) (d)

FIGURE 2.1 – Caractéristiques géométriques, (a) Rectangularité, (b) Convexité, (c) Compacité, (d) Excentricité.

2.2.1.4 Les moments statistiques

Pour une image binaire définie sur un support $I = (n \times m)$, les moments statistiques ou spatiaux de degré (p, q) sont définis par ([Pra01]) :

$$m_{pq} = \sum_{(x,y)\in I} x^p y^q f(x,y),\ \forall (p,q) \in \mathbb{N} \tag{2.25}$$

L'ordre de m_{pq} est donné par $p + q$, sa répétition par $|p - q|$. Les moments statistiques normés sont indépendants du facteur d'échelle de l'image :

$$m_{pq}^{norm} = \frac{1}{n^p m^q} \sum_{(x,y)\in I} x^p y^q f(x,y),\ \forall (p,q) \in \mathbb{N} \tag{2.26}$$

Les moments statistiques centrés sont, quant à eux, invariants par translation :

$$\mu_{pq} = \sum_{(x,y)\in I} (x - \bar{x})^p (y - \bar{y})^q f(x,y),\ \forall (p,q) \in \mathbb{N} \tag{2.27}$$

avec $\bar{x} = \frac{m_{10}}{m_{00}}$ et $\bar{y} = \frac{m_{01}}{m_{00}}$. Enfin les moments statistiques centrés normés possèdent la double propriété d'être invariants par changement d'échelle et par translation :

$$\mu_{pq}^{norm} = \frac{1}{n^p m^q} \sum_{(x,y)\in I} (x - \bar{x})^p (y - \bar{y})^q f(x,y),\ \forall (p,q) \in \mathbb{N} \tag{2.28}$$

2.2.1.5 Les moments de Hu

HU ([Hu62]) a défini, à partir des moments statistiques centrés, un jeu de descripteurs invariants par rotation.

Considérons la normalisation suivante :

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{(p+q+2)/2}} \tag{2.29}$$

avec $p+q \geq 2$ et $(p,q) \in \mathbb{N}$. Les 7 moments de Hu sont définis à partir des η_{pq} d'ordre 2 et 3 :

$$\begin{array}{ll}
\phi_1 = & \eta_{20} + \eta_{02} \\
\phi_2 = & (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
\phi_3 = & (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
\phi_4 = & (\eta_{30} + \eta_{12})^2 + (\eta_{21} - \eta_{03})^2 \\
\phi_5 = & (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12}) \left[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2\right] + \\
& (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03}) \left[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right] \\
\phi_6 = & (\eta_{20} - \eta_{02})((\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2) + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \\
\phi_7 = & (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12}) \left[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2\right] - \\
& (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03}) \left[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right]
\end{array} \tag{2.30}$$

2.2.1.6 Les moments complexes et les invariants de Flusser et Suk.

Pour évaluer la qualité des moments invariants de HU, ABU-MOSTAFA ET PSALTIS ([AMP84]) introduisent les moments complexes. L'expression discrète, non centrée, non normée des moments complexes de degré (p,q) s'exprime, pour une image de support $I = (n \times m)$, selon l'équation :

$$c_{pq} = \sum_{(x,y)\in I} (x+iy)^p (x-iy)^q f(x,y),\ \forall (p,q) \in \mathbb{N} \tag{2.31}$$

où i représente l'unité complexe. L'ordre de c_{pq} est donné par $p+q$, sa répétition par $|p-q|$. FLUSSER ET SUK ([FS03]) donnent l'expression des moments complexes en fonction des moments statistiques :

$$c_{pq} = \sum_{k=0}^{p} \sum_{j=0}^{q} C_k^p C_j^q (-1)^{(q-j)} . i^{(p+q-k-j)} . m_{(k+j,p+q-k-j)} \tag{2.32}$$

avec C_k^p le nombre de combinaisons de k éléments parmi p, et inversement :

$$m_{pq} = \frac{1}{2^{(p+q)} i^q} \sum_{k=0}^{p} \sum_{j=0}^{q} C_k^p C_j^q (-1)^{(q-j)} . i^{(p+q-k-j)} . c_{(k+j,p+q-k-j)} \tag{2.33}$$

Comme pour les moments statistiques, une version centrée invariante en translation des moments complexes s'obtient par :

$$\gamma_{pq} = \sum_{(x,y)\in I} (x + iy - \overline{(x+iy)})^p (x - iy - \overline{(x-iy)})^q f(x,y),\ \forall (p,q) \in \mathbb{N} \tag{2.34}$$

avec :

$$\left\{ \begin{array}{l} \overline{(x+iy)} = \bar{x} + i\bar{y} = \frac{c_{10}}{c_{00}} = \frac{m_{10} + i m_{01}}{m_{00}} \\ \overline{(x-iy)} = \bar{x} - i\bar{y} = \frac{c_{01}}{c_{00}} = \frac{m_{10} - i m_{01}}{m_{00}} \end{array} \right. \tag{2.35}$$

De même que pour les moments statistiques, les $\{\frac{\gamma_{pq}}{(n+im)^p (n-im)^q}\}$ sont invariants par translation et changement d'échelle. En exprimant les moments complexes en coordonnées polaires, ABU-MOSTAFA ET PSALTIS ([AMP84]) montrent facilement que les $|c_{pq}|$ sont invariants par rotation. Cependant comme

$c_{pq} = c^{\star}_{qp}$, où * représente le complexe conjugué, les $|c_{pq}|$ ne permettent d'obtenir qu'un ensemble de $\frac{N}{2}+1$ invariants à partir d'un ensemble original de $N+1$ invariants d'ordre N. Il y a donc perte d'information. De plus, ils montrent que les attributs d'une image correspondant aux variations angulaires de $N+1$ cycles/cycle et plus sont complètement perdus si les moments sélectionnés sont d'ordre au plus $(N+1)(N+2)/2$, ceci est valable également pour les moments de Zernike que nous verrons plus loin.

A partir des moments complexes, il est possible d'exprimer les moments de HU. Considérons la normalisation suivante :

$$\vartheta_{pq} = \frac{\gamma_{pq}}{\gamma_{00}^{(p+q+2)/2}} \tag{2.36}$$

avec $p+q \geq 2$ et $(p,q) \in \mathbb{N}$. Les 7 moments de Hu de l'équation 2.30 page précédente peuvent alors s'exprimer en fonction des ϑ_{pq} :

$$\begin{aligned}
\phi_1 &= \vartheta_{11} \\
\phi_2 &= \vartheta_{20}\vartheta_{02} \\
\phi_3 &= \vartheta_{30}\vartheta_{03} \\
\phi_4 &= \vartheta_{21}\vartheta_{12} \\
\phi_5 &= Re(\vartheta_{30}\vartheta_{12}^3) \\
\phi_6 &= Re(\vartheta_{20}\vartheta_{12}^2) \\
\phi_7 &= Im(\vartheta_{30}\vartheta_{12}^3)
\end{aligned} \tag{2.37}$$

Généralisant cette formulation, FLUSSER & SUK ([Flu00], [FS03]) proposent une méthode pour obtenir un jeu complet de moments invariants par rotation à partir des c_{pq} et du théorème 1 :

Théorème 1 *Soit $n \geq 1$ et $(k_i, p_i, q_i) \in \mathbb{N}$ avec $i = (1, \ldots, n)$ et tels que :*

$$\sum_{i=1}^{n} k_i(p_i - q_i) = 0 \tag{2.38}$$

alors tous les $\mathfrak{I}$ définis de la manière suivante :

$$\mathfrak{I} = \prod_{i=1}^{n} c_{p_i q_i}^{k_i} \tag{2.39}$$

sont invariants par rotation.

Le théorème 1 permet de construire une infinité d'invariants pour un ordre donné mais seulement quelques-uns d'entre eux sont indépendants mutuellement. Il convient d'en extraire une base, c'est à dire un ensemble complet de moments indépendants. Pour ce faire FLUSSER & SUK expriment et prouvent le théorème 2 :

Théorème 2 *Soit l'ensemble des moments complexes d'ordre inférieur ou égal à $r \geq 2$. Soit l'ensemble des invariants $\mathfrak{B} = \{\Phi_{pq} | \forall (p,q),\ p \geq q \wedge p+q \leq r\}$ définis de la manière suivante :*

$$\Phi_{pq} = c_{pq} c_{q_0 p_0}^{p-q} \tag{2.40}$$

avec q_0 et p_0 choisis arbitrairement tels que $q_0 + p_0 \leq r$, $p_0 - q_0 = 1$ et $c_{p_0 q_0} \neq 0$ pour les images traitées. Alors $\mathfrak{B}$ forme une base d'invariants par rotation.

En partant de la constatation que les moments de Hu tels que présentés dans l'équation 2.37 ne sont clairement pas indépendants :

$$\phi_3 = \vartheta_{30}\vartheta_{03} = \frac{\vartheta_{03}\vartheta_{21}^3\vartheta_{30}\vartheta_{12}^3}{(\vartheta_{21}\vartheta_{12})^3} = \frac{\phi_5^2 + \phi_7^2}{\phi_4^3} \tag{2.41}$$

FLUSSER & SUK ([Flu00]) explicitent, à partir du théorème 2, une base du deuxième, troisième et quatrième ordre de moments invariants par rotation. En utilisant la normalisation de HU et les moments

complexes centrés, une version invariante par translation, changement d'échelle et rotation des moments de FLUSSER & SUK est définie par (équation 2.42) :

$$\begin{aligned}
\psi_1 &= \vartheta_{11} = \phi_1, \\
\psi_2 &= \vartheta_{21}\vartheta_{12} = \phi_4, \\
\psi_3 &= Re(\vartheta_{20}\vartheta_{12}^2) = \phi_6, \\
\psi_4 &= Im(\vartheta_{20}\vartheta_{12}^2) \\
&= \eta_{11}((\eta_{30}+\eta_{12})^2 - (\eta_{03}+\eta_{21})^2) \\
&\quad -(\eta_{20}-\eta_{02})(\eta_{30}+\eta_{12})(\eta_{03}+\eta_{21}), \\
\psi_5 &= Re(\vartheta_{30}\vartheta_{12}^3) = \phi_5, \\
\psi_6 &= Im(\vartheta_{30}\vartheta_{12}^3) = \phi_7, \\
\psi_7 &= \vartheta_{22}, \\
\psi_8 &= Re(\vartheta_{31}\vartheta_{12}^2), \\
\psi_9 &= Im(\vartheta_{31}\vartheta_{12}^2), \\
\psi_{10} &= Re(\vartheta_{40}\vartheta_{12}^4), \\
\psi_{11} &= Im(\vartheta_{40}\vartheta_{12}^4),
\end{aligned} \tag{2.42}$$

2.2.1.7 Les moments de Zernike

Les polynômes de Zernike ont été définis en 1934 dans le cadre de la théorie de la diffraction optique ([Zer34]). Dérivés de ces polynômes, les moments ont été utilisés par de nombreux auteurs en reconnaissance de caractères ([Tea80], [KH90a], [KH90b], [BSA91]). Plusieurs études montrent également la supériorité de ces descriptions par rapport à d'autres approches ([TC88], [KK00]).

Les moments A_{nm} de Zernike correspondent à la projection de la forme $f(x,y)$ sur une base ZP de fonctions orthogonales $V_{nm}(x,y)$ et se définissent par :

$$A_{nm} = \frac{n+1}{\pi} \sum_{(x,y)\in I} V_{nm}(x,y)^* f(x,y) \tag{2.43}$$

* définissant le complexe conjugué. La base ZP est définie sur le cercle unité par :

$$ZP = \{V_{nm}(x,y) \mid (x^2+y^2) \leq 1\} \tag{2.44}$$

où le polynôme complexe V_{nm} d'ordre n et de répétition m est défini avec $n \in \mathbb{N}^+$ et $m \in \mathbb{N}$ tel que $n-|m|$ soit pair et $|m| \leq n$:

$$V_{nm}(x,y) = R_{nm}(r)e^{jm\theta} \text{ en coordonnées polaires} \tag{2.45}$$

avec :

$$R_{nm}(r) = \sum_{s=0}^{(n-|m|)/2} (-1)^s \frac{(n-s)!r^{n-2s}}{s!\left(\frac{n+|m|}{2}-s\right)!\left(\frac{n-|m|}{2}-s\right)!} \tag{2.46}$$

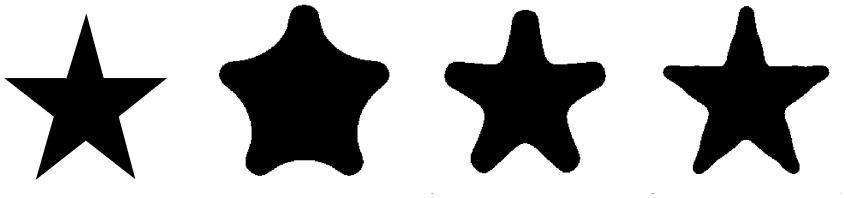
Les moments sont invariants par rotation, translation et changement d'échelle (après normalisation de la taille de la forme). De plus, grâce à l'exploitation d'une base de fonctions orthogonales, ces moments sont peu corrélés.

Cette représentation est inversible, l'image peut être reconstruite de la manière suivante :

$$\hat{f}(x,y) = \lim_{N\to+\infty} \sum_{n=0}^{N} \sum_{m} A_{nm} V_{nm}(x,y) \tag{2.47}$$

L'ordre des moments possède une grande influence sur la conservation de l'information angulaire. Plus l'ordre est élevé et plus les variations angulaires décrites sont fines. La figure 2.2 page suivante en donne une illustration.

Plusieurs stratégies d'implémentation ont été développées ([MR95], [BAS96]). CHONG & AL ([CRM03]) présentent une étude comparative à ce sujet.



(a) (b) (c) (d)

FIGURE 2.2 – Exemple de reconstructions à partir des descripteurs de Zernike, (a) image d'origine, (b) reconstruction d'ordre 10, (c) reconstruction d'ordre 20, (d) reconstruction d'ordre 40.

2.2.2 Descripteurs externes

Comme pour une forme, l'image I d'un contour peut être vue comme une application $g(x, y) \rightarrow \{0, 1\}$ où $(x, y) \in I$ sont les coordonnées des pixels de l'image. Si le pixel (x, y) est en dehors du contour $g(x, y) = 0$ et $g(x, y) = 1$ s'il appartient au contour.

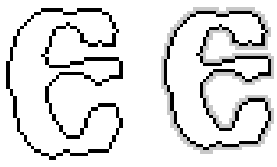
2.2.2.1 Extraction de contour

Il y a plusieurs méthodes pour obtenir le contour d'une forme binaire. Les plus simples sont les méthodes morphologiques (voir l'annexe C page 169). Le contour externe, par exemple, s'obtient en dilatent la forme à partir d'un masque 3×3 plein puis en lui soustrayant l'image de départ (figure 2.3).

(a) (b) (c)

FIGURE 2.3 – Extraction morphologique de contour externe, (a) image originale, (b) image dilatée, (c) contour externe.

Il est également possible d'utiliser une méthode à base de gradient de type filtre de Sobel par exemple (figure 2.4).



(a) (b)

FIGURE 2.4 – Extraction par filtrage du contour, (a) image du contour par filtrage de Sobel, (b) superposition des deux images de contours (contour morphologique externe en gris).

2.2.2.2 Les descripteurs de Fourier pour contours fermés

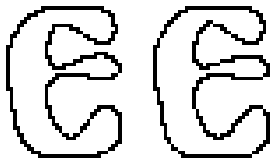
Les descripteurs de Fourier sont obtenus à partir de la décomposition en série de Fourier d'une grandeur unidimensionnelle, appelée signature, extraite du contour.

Echantillonnage Plusieurs signatures du contour d'une forme peuvent être extraites. La plus courante est la distance au centre de gravité de la courbe ([PF77]), cependant les positions complexes ([Gra72]) ou les variations angulaires ([ZR72]) sont également de bonnes candidates.

Toutes ces techniques font appel à un échantillonnage de la courbe soit angulaire, soit linéique. Cet échantillonnage doit être réalisé à pas constant pour rester dans le cas unidimensionnel. De plus, un échantillonnage angulaire est impossible sur les courbes non convexes.

Pour les contours 4-connexes comme sur la figure 2.3 page précédente (c) l'échantillonnage à longueur constante est simple à réaliser puisqu'il suffit de considérer tous les points de la courbe qui sont par définition espacés deux à deux par une longueur constante de 1. En revanche pour les courbes 8-connexes comme c'est le cas sur la figure 2.4 page ci-contre (a), l'espacement des points deux à deux vaut soit 1 soit $\sqrt{2}$, il faut donc *a priori* rééchantillonner.

En pratique, sur des images de caractères extraites de pages manuscrites acquises à faible résolution (300 dpi), l'erreur introduite par l'imprécision de l'échantillonnage est à peu près du même ordre que celle induite par la prise en compte de tous les points de contour. La figure 2.5 montre une reconstruction effectuée à partir d'une description de Fourier sur les positions complexes du contour de la figure 2.4 page ci-contre (a). La qualité des deux reconstructions est comparable et l'échantillonnage facultatif.



(a) (b)

FIGURE 2.5 – Reconstructions à partir de 20 descripteurs complexes de Fourier, (a) sans échantillonnage, (b) échantillonnage de 64 points à longueur constante.

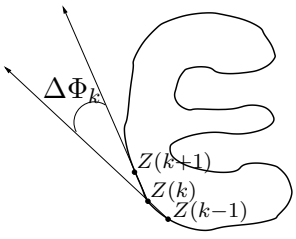


FIGURE 2.6 – Descripteurs de Fourier angulaires.

Signature angulaire ZAHN ET ROSKIES ([ZR72]) utilisent la différence angulaire segment à segment d'un contour décrit dans le sens anti-trigonométrique (figure 2.6 page précédente). Les descripteurs suivants peuvent être calculés :

$$a_n = -\frac{1}{n\pi}\sum_{k=1}^{m}\Delta\Phi_k \sin(\frac{2\pi nk}{m}) \tag{2.48}$$

$$b_n = \frac{1}{n\pi}\sum_{k=1}^{m}\Delta\Phi_k \cos(\frac{2\pi nk}{m}) \tag{2.49}$$

où m est le nombre de segments décrivant le contour et $\Delta\Phi_k$ correspond à la différence angulaire des segments $[Z(k+1), Z(k)]$ et $[Z(k), Z(k-1)]$. Dans le cas d'un contour 8-connexe, $\Delta\Phi_k$ peut également être remplacée par la courbure $\kappa(t) = \Delta\Phi_k/\Delta(t)$. Les a_n et b_n sont invariants par changement d'échelle et translation. L'invariance par rotation s'obtient, quant à elle, en utilisant les amplitudes des moments complexes. Ainsi dans l'équation 2.50 A_n est invariant par rotation alors que la phase α_n dans l'équation 2.51 ne l'est pas.

$$A_n = \sqrt{a_n^2 + b_n^2} \tag{2.50}$$

$$\alpha_n = \tan(a_n/b_n) \tag{2.51}$$

Zahn et Roskies définissent aussi un jeu de descripteurs invariants par rotation mais pas par effet de miroir, ce qui peut-être intéressant pour distinguer un 'p' d'un 'q' par exemple :

$$F_{kj} = j^* \times \alpha_k - k^* \times \alpha_j \tag{2.52}$$

où $j^* = j/pgcd(j,k)$ et $k^* = k/pgcd(j,k)$, *pgcd* signifiant « plus grand commun diviseur ».

Typiquement, la signature angulaire présente deux inconvénients. Tout d'abord elle est indéfinie si l'amplitude A_n est nulle (variation angulaire nulle), ensuite, elle peut comporter des discontinuités génératrices d'imprécisions.

Descriptions complexes ([PF77],[Gra72])

La position complexe est simplement le nombre complexe généré par les coordonnées des points de contour :

$$s_k = (x_k - x_c) + i(y_k - y_c) \tag{2.53}$$

où (x_c, y_c) sont les coordonnées du centre de gravité de la courbe. La transformée de Fourier discrète (DFT) de s_k s'écrit :

$$a_u = \frac{1}{n\pi}\sum_{k=1}^{m} s_k exp^{-\frac{j2\pi uk}{m}} \tag{2.54}$$

avec $-m \leq u \leq m$. Les descripteurs f_Z sont obtenus par :

$$f_Z = \left[\frac{|a_{-m/2-1}|}{|a_1|}, \ldots, \frac{|a_{-1}|}{|a_1|}, \frac{|a_2|}{|a_1|}, \ldots, \frac{|a_{m/2}|}{|a_1|}\right] \tag{2.55}$$

Ils sont invariants par translation (les coordonnées sont relatives au centre de gravité), changement d'échelle (la composante continue f_0 est la seule sensible aux changements d'échelles) et rotation (seules les amplitudes sont utilisées).

Distance au centre de gravité ([PF77])

La distance des points de la courbe au centre de gravité (x_c, y_c) est établie en chaque point par l'équation :

$$R(t) = \sqrt{[x(t) - x_c]^2 + [y(t) - y_c]^2} \tag{2.56}$$

De la DFT (équation 2.57), seul le module est conservé au vu de son invariance en rotation. Les fréquences négatives ne sont pas non plus conservées puisque $R(t)$ est à valeurs réelles, les coefficients fréquentiels sont symétriques par rapport à la composante continue.

$$F_i = \frac{1}{m} \sum_{t=0}^{m-1} R(t) exp(\frac{-j2\pi it}{m}) \; i = 0, 1, \cdots, m-1 \tag{2.57}$$

Le vecteur d'attributs ainsi défini possède des coefficients normalisés par $|F_0|$ pour s'affranchir des changements d'échelle :

$$f_R = \left\{ \frac{|F_1|}{|F_0|}, \frac{|F_2|}{|F_0|}, \cdots, \frac{|F_{\frac{m}{2}}|}{|F_0|} \right\} \tag{2.58}$$

Au final les f_R sont invariants par translation (distance au centre de gravité), rotation (seul le module est conservé) et changement d'échelle (normalisation par la composante continue).

Longueur de Corde Si le centre de gravité se trouve en dehors de la courbe, le profil de la distance entre le centre et la courbe peut être très différent pour deux formes proches. ZHANG & LU ([ZL04]) ont proposé d'utiliser une distance entre deux points opposés de la courbe.

Prenons le point P de la figure 2.7, $r^\star(t)$ est la distance de P à un autre point de la courbe P' tel que (PP') soit perpendiculaire à la tangente à la courbe en P. Si la droite (PP') passe par d'autres points de la courbe (P_1 par exemple) le point est choisi tel que le milieu du segment $[PP']$ soit à l'intérieur de la courbe. Ainsi P_1 ne pourra être choisi car P_2 est à l'extérieur de la courbe. Imaginons que P_2 soit, lui aussi, à l'intérieur de la courbe alors il faudra considérer les milieux de $[PP_2]$ et $[P_2P_1]$ et ainsi de suite jusqu'à trouver ou non un milieu de segment en dehors de la courbe. Si un tel point est trouvé, P_1 sera rejeté.

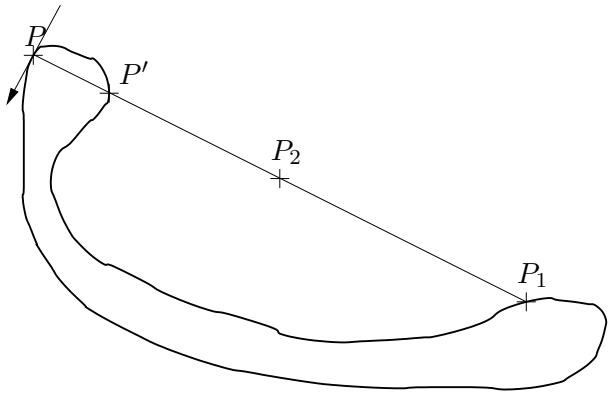


FIGURE 2.7 – Descripteurs de Fourier suivant la longueur de corde.

Les descripteurs $f_{r^\star}$ se calculent de la même manière que les f_R (équations 2.57 et 2.58) et ont les mêmes propriétés d'invariance. ZHANG & LU soulignent toutefois que $r^\star(t)$ est très sensible aux approximations numériques notamment sur le calcul des angles. Ils conseillent d'utiliser un filtrage médian en amont pour réduire les aspérités de la courbe.

2.2.2.3 Les descripteurs de Fourier pour contours ouverts

Pour résoudre le problème des contours ouverts, RAUBER & STEIGER ([RSG92]) ont proposé une famille de descripteurs de Fourier capablent de décrire les courbes ouvertes et les lignes : les descripteurs UNL[2].

Le calcul se fait en deux étapes. La première (transformation UNL) transforme les coordonnées cartésiennes des couples de points voisins deux à deux en coordonnées polaires centrées sur le centre de gravité de la courbe. Puis est appliquée une transformée de Fourier bidimensionnelle de l'image.

Soit $\Omega(t)$ la courbe ouverte composée de n pixels $z_i = (x_i, y_i)$, $O = (O_x, O_y)$ le centre de gravité de la courbe et M la distance maximale entre O et les z_i. Soit $z_{ij}(t)$ le segment entre $z_i = (x_i, y_i)$ et $z_j = (x_j, y_j)$ deux points voisins de la courbe, le point $U(z_{ij}(t))$ de l'image transformée suivant la transformation UNL s'obtient par :

$$U(z_{ij}(t)) = (E_{ij}(t), \theta_{ij}(t)) = \left(\frac{\|z_i + t(z_j - z_i) - O\|}{M}, \arctan\left(\frac{y_i + t(y_j - y_i) - O_y}{x_i + t(x_j - x_i) - O_x} \right) \right) \quad (2.59)$$

Ce qui donne l'image en coordonnées polaires $f(R, \Theta)$. Son spectre sans la composante continue et les harmoniques à conjuguées symétriques donnent les descripteurs $UFF(u', v')$ après normalisation par la composante continue :

$$UFF(u', v') = \frac{\|\mathcal{F}(f(R, \Theta))\|}{F(0,0)} = \frac{\|F(u, v)\|}{F(0,0)} \quad (2.60)$$

2.3 Calcul de distance

Une fois l'image de la forme décrite par un vecteur de caractéristiques, se pose la question de la comparaison des formes et donc des vecteurs. Nous avons vu au paragraphe 2.1 page 20 que l'outil mathématique de calcul de proximité était la distance ou une similarité associée. Si nous disposons de n descripteurs par forme, l'espace vectoriel de définition est $\mathbb{R}^n \times \mathbb{R}^n$ et nous disposons de toutes les distances définies sur $\mathbb{R}^n$ pour comparer nos formes. La plus courante étant bien sûr la *distance euclidienne* :

$$d_E(\mathbf{x}, \mathbf{y}) = \Big((\mathbf{x} - \mathbf{y})^{\perp} (\mathbf{x} - \mathbf{y}) \Big)^{\frac{1}{2}} \ , \ \forall (\mathbf{x}, \mathbf{y}) \in \mathbb{R}^n \times \mathbb{R}^n \quad (2.61)$$

La distance euclidienne est un cas particulier des *distances de Minkowski* :

$$d_{Mink}(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{n} |\mathbf{x}(i) - \mathbf{y}(i)|^p \right)^{\frac{1}{p}} \ , \ \forall (\mathbf{x}, \mathbf{y}) \in \mathbb{R}^n \times \mathbb{R}^n \ , \ p \in \mathbb{N}^{+\star} \quad (2.62)$$

Toutes ces distances comparent deux vecteurs attribut par attribut. Le problème est que les domaines de définition de chacun des attributs ne sont pas identiques. La rectangularité, par exemple, est définie sur $[0, 1]$ alors que les moments statistiques le sont sur $\mathbb{R}_{\star}^{+}$. L'impact des attributs sera donc différent lors du calcul des distances de Minkowski. Une solution possible consiste à pondérer ces distances en fonction des caractéristiques de l'attribut, ce qui donne pour la distance euclidienne :

$$d_{EW}(\mathbf{x}, \mathbf{y}) = \Big((\mathbf{x} - \mathbf{y})^{\perp} W (\mathbf{x} - \mathbf{y}) \Big)^{\frac{1}{2}} \quad (2.63)$$

où W est une matrice diagonale de poids. Le carré de cette distance est aussi appelé *distance quadratique*. Les poids pourraient correspondre intuitivement à une simple normalisation par rapport à l'ensemble de définition de l'attribut. Malheureusement il ne donne aucune garantie sur la distribution des

[2] Universidade Nova de Lisboa

Distance	Définition	f	g
Distance de Chernoff	$d_C(p,q) = -\log(\sum_i p_i^r q_i^{1-r})$	$-x^{1-r}$	$-\log(-x)$
Distance de Bhattacharyya	$d_B(p,q) = -\log(\sum_i (p_i q_i)^{\frac{1}{2}})$	Comme Chernoff avec $r = \frac{1}{2}$	
Distance de Matusita	$d_{MT}(p,q) = (\sum_i (\sqrt{p_i} + \sqrt{q_i})^2)^{\frac{1}{2}}$	$\lvert 1 - \sqrt{x}\rvert^2$	$\sqrt{x}$
Divergence de Kullback	$d_K(p,q) = \sum_i (p_i \log \frac{p_i}{q_i})$	$-\log x$	x
Divergence symétrique de Kullback	$d_{KS}(p,q) = \sum_i (p_i - q_i) \log \frac{p_i}{q_i}$	$(x-1)\log x$	x

TABLE 2.1 – Quelques distances probabilistes, avec $0 < r < 1$ ([Bas88])

valeurs à l'intérieur de cet ensemble et il est plus judicieux, par exemple et quand elle est connue, de normaliser par la variance. C'est la distance de Mahalanobis :

$$d_M(\mathbf{x},\mathbf{y}) = \Big((\mathbf{x}-\mathbf{y})^{\perp}\Sigma^{-1}(\mathbf{x}-\mathbf{y})\Big)^{\frac{1}{2}} \tag{2.64}$$

où Σ représente la matrice de covariance connue *a priori*. L'utilisation de la matrice de covariance nous place implicitiment dans le cadre formel des approches probabilistes. La distance de Mahalanobis peut s'interpréter plus généralement comme la distance entre deux lois de probabilités multidimensionnelles gaussiennes de variance identique Σ et centrées respectivement en $\mathbf{x}$ pour la première et en $\mathbf{y}$ pour la seconde. Plus généralement, le cadre probabiliste a permis de définir d'autres distances définies entre deux lois de probabilités. C'est le cas des *f-divergences*, définies par CSISZAR ([Csi63]) et ALI ET SILVEY ([AS66]) :

Définition 5 (f-divergence) *Soit $\Omega = \{\mathbf{x_1}, \ldots, \mathbf{x_n}\}$ un ensemble d'au moins deux éléments et $\mathcal{P}$ l'ensemble de toutes les distributions de probabilités $p = \{p_i | p_i = Pr(\mathbf{x_i}), \mathbf{x_i} \in \Omega\}$. Soient f une fonction convexe*[3] *$f : [0,+\infty[\to \mathbb{R}$ continue en 0* [4]*, g une fonction croissante sur $\mathbb{R}$ et une paire $(p,q) \in \mathcal{P}$, alors :*

$$I_f(p,q) = g\left(\sum_{i=1}^{n} q_i f\left(\frac{p_i}{q_i}\right)\right) \tag{2.65}$$

est appelée une f-divergence *des distributions de probabilité p et q.*

La table 2.1 donne quelques distances probabilistes qui sont des f-divergences tirées de [Bas88]. Quelques propriétés intéressantes sont à noter concernant ces distances : la distance de Bhattacharyya est un cas particulier de la distance de Chernoff (avec $r = 1/2$) ; la distance de Matusita est corrélée à celle de Bhattacharrya avec la relation $d_{MT} = \{2[1 - exp(-d_B)]\}^{1/2}$; la divergence symétrique de Kullback correspond à la version symétrique de la divergence de Kullback[5] $d_{KS}(p,q) = d_K(p,q) + d_K(q,p)$.

Sous une hypothèse de normalité multivariée, si $p \rightsquigarrow \mathcal{N}(\mu_p, \Sigma_p)$ et $q \rightsquigarrow \mathcal{N}(\mu_q, \Sigma_q)$ alors les distances de la table 2.1 s'expriment analytiquement comme dans la table 2.2 page suivante (les expressions sont tirées de [ZC06]). Notons que pour $\Sigma_p = \Sigma_q = \Sigma$, la distance de Mahalanobis s'écrit : $d_M^2(p,q) = d_{KS}(p,q) = 8 \times d_B(p,q)$.

Il existe encore d'autres distances probabilistes utilisées en classification et indexation ([HMMG01]), mais toutes nécessitent une estimation des lois de probabilités soit sous forme d'histogrammes des fréquences ou soit sous forme d'hypothèses sur les distributions qui sont bien souvent gaussiennes. Elles n'ont pas d'intérêt dans la cadre de la comparaison de deux vecteurs numériques. Par contre elles servent à calculer la distance entre deux séries de vecteurs considérés alors comme les observations de deux

[3] c.-à-d. $\forall \lambda \in [0,1]$ et $\forall (x,y) \in [0,+\infty[^2$, $f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y)$

[4] c.-à-d. $f(0) = \lim_{x \to 0} f(x)$

[5] aussi appelée entropie relative

Distance	Expression analytique
Chernoff	$d_C(p,q) = \frac{1}{2}r(1-r)(\mu_p-\mu_q)^{\perp}[r\Sigma_p+(1-r)\Sigma_q]^{-1}(\mu_p-\mu_q) + \frac{1}{2}\log\frac{\lvert r\Sigma_p+(1-r)\Sigma_q\rvert}{\lvert\Sigma_p\rvert^r\lvert\Sigma_q\rvert^{1-r}}$
Bhattacharyya	$d_B(p,q) = \frac{1}{8}(\mu_p-\mu_q)^{\perp}[\frac{1}{2}(\Sigma_p+\Sigma_q)]^{-1}(\mu_p-\mu_q) + \frac{1}{2}\log\frac{\frac{1}{2}\lvert\Sigma_p+\Sigma_q\rvert}{\lvert\Sigma_p\rvert^{1/2}\lvert\Sigma_q\rvert^{1/2}}$
Matusita	se déduit par : $d_{MT}(p,q) = \{2[1-exp(-d_B(p,q))]\}^{1/2}$
Kullback	$d_K(p,q) = \frac{1}{2}(\mu_p-\mu_q)^{\perp}\Sigma_q^{-1}(\mu_p-\mu_q) + \frac{1}{2}\log\frac{\lvert\Sigma_q\rvert}{\lvert\Sigma_p\rvert} + \frac{1}{2}\mathrm{tr}[\Sigma_p\Sigma_q^{-1}-Id]$
Kullback Sym.	$d_{KS}(p,q) = \frac{1}{2}(\mu_p-\mu_q)^{\perp}(\Sigma_p^{-1}+\Sigma_q^{-1})(\mu_p-\mu_q) + \frac{1}{2}\mathrm{tr}[\Sigma_p\Sigma_q^{-1}+\Sigma_p^{-1}\Sigma_q-2Id]$

TABLE 2.2 – Expressions analytiques des distances probabilistes de la table 2.1 page précédente entre deux densités normales multivariées.

vecteurs aléatoires.

Munis de descripteurs appropriés et d'une distance, nous pouvons maintenant définir un sytème probabiliste pour la reconnaissance statistique de caractères.

CHAPITRE 3

SYSTÈMES PROBABILISTES DE RECONNAISSANCE STATISTIQUE

Sommaire

Dans un espace d'attributs muni d'une métrique, la reconnaissance consiste à attribuer au vecteur décrivant une forme, une valeur linguistique (sa signification en tant que caractère). Pour ce faire, il faut diviser l'espace des représentations en régions clairement différentiables qui seront chacune associée à une et une seule classe par une application appelée *fonction d'étiquetage*. La division de l'espace est la phase d'*apprentissage*. Elle s'effectue la plupart du temps à partir d'une *base d'apprentissage* qui est un ensemble de données étiquetées. Nous parlerons par la suite de *classification* qui, lorsque les données sont étiquetés, est dite *supervisée*. Cette étape d'apprentissage peut être précédée d'une réduction de la dimension de l'espace afin de réduire la complexité calculatoire. Une fois l'apprentissage terminé, la reconnaissance consiste à décider de l'étiquette à attribuer à tout nouvel individu en fonction de sa position dans l'espace des représentations.

Formellement, la base d'apprentissage est un doublet $\mathcal{B} = (\mathcal{O}, \epsilon)$ où $\mathcal{O} = \{\mathbf{x}_i\}$, $i = 1, \ldots, m$, est l'ensemble des observations définies sur $\mathbb{R}_n$ décrivant les formes et :

$$\begin{array}{rl} \epsilon \ : & \mathbb{R}^n \rightarrow \mathcal{E} \\ & \mathbf{x} \rightarrow \omega_k \end{array} \tag{3.1}$$

est la fonction d'étiquetage. Les ω_k, $k = 1, \ldots, c$, sont les étiquettes des classes avec $c = \mathrm{Card}(\mathcal{E})$ le nombre total de classes (une classe n'est associée qu'à une seule étiquette). La reconnaissance consiste alors à affecter à une nouvelle observation non étiquetée, une étiquette de classe par comparaison avec $\mathcal{B}$.

Ce chapitre rappelle le formalisme de méthodes statistiques et probabilistes de reconnaissance qui seront utilisées dans le chapitre suivant. Les méthodes exposées ici sont simples et classiques mais il nous est paru utile de les rappeler afin, d'une part, d'introduire les notations et surtout, d'autre part, les différents paramètres qui seront utiles par la suite. Nous commencerons par rappeler une technique de réduction de données, puis nous exposerons quelques approches probabilistes de classification et plus particulièrement de classification supervisée. Enfin nous expliciterons la stratégie de décision que nous avons utilisée dans les différents tests.

3.1 La réduction de données

Pour résoudre le problème du fléau de la dimension, qui veut que dans un espace de trop grande dimension tous les objets soient voisins, une étape de réduction de données est généralement mise en œuvre afin de réduire la dimension de l'espace. La méthode la plus courante est l'*Analyse en Composantes Principales* ou ACP construite sur le principe de la maximisation de la variance dans l'espace de projection ([Hot33]). L'ACP dans sa forme première est une méthode linéaire factorielle reposant sur la diagonalisation de la matrice de covariance.

Le formalisme et l'implémentation de l'ACP sont relativement simples et c'est pourquoi ils sont exploités dans de nombreux domaines. C'est, dans sa forme originale, une méthode d'analyse exploratoire de données. Elle est basée sur le même principe que la transformée de Karhunen-Loeve (paragraphe 2.2.1.2 page 22). Considérons un jeu de données $\{\mathbf{x_i}\},\ i = 1, \ldots, m$ vecteurs de $\mathbb{R}^n$. La matrice des données centrées s'écrit $X = [(\mathbf{x_1} - \bar{\mathbf{x}}) \ldots (\mathbf{x_m} - \bar{\mathbf{x}})]$ et la matrice de covariance s'estime par :

$$\bar{\Sigma}_X = XPX^{\perp} \tag{3.2}$$

où P est la matrice des poids :

$$P = \begin{bmatrix} \frac{1}{m} & 0 & \ldots & 0 \\ 0 & \frac{1}{m} & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & \frac{1}{m} \end{bmatrix} \tag{3.3}$$

$\bar{\Sigma}_X$ est réelle et symétrique donc diagonalisable et s'écrit :

$$\bar{\Sigma}_X = W\Lambda W^{\perp} = [\mathbf{w_1} \ldots \mathbf{w_n}] \begin{bmatrix} \lambda_1 & 0 & \ldots & 0 \\ 0 & \lambda_2 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & \lambda_n \end{bmatrix} \begin{bmatrix} \mathbf{w_1}^{\perp} \\ \mathbf{w_2}^{\perp} \\ \ldots \\ \mathbf{w_n}^{\perp} \end{bmatrix} \tag{3.4}$$

où Λ est la matrice diagonale des valeurs propres et W est la matrice $n \times n$ des vecteurs propres $\mathbf{w_j}$. La matrice W est une matrice de rotation puisque $W^{\perp}W = I$. Ces vecteurs propres définissent une nouvelle base dans laquelle les données de départ peuvent être projetées :

$$\mathbf{y_i} = W^{\perp}(\mathbf{x_i} - \bar{\mathbf{x}}) \tag{3.5}$$

Les $\{\mathbf{y_i}\},\ i = 1, \ldots, m$ ont Λ comme estimation de la matrice de covariance ce qui signifie que dans cette nouvelle base les différentes composantes sont décorrélées et que les λ_j sont leurs variances. Le vecteur propre correspondant à la plus grande valeur propre est la *composante principale* des données.

En rangeant les vecteurs propres dans l'ordre décroissant de leurs valeurs propres ($\lambda_1^{\star} \geq \lambda_2^{\star} \geq \ldots \geq \lambda_n^{\star}$), nous obtenons une nouvelle matrice $W^{\star}$ de transformation des données initiales. Chacun des axes de ce repère possède une *inertie* définie par :

$$\mathcal{I}_j = \frac{\lambda_j^{\star}}{\sum_{k=1}^{n} \lambda_k^{\star}} \tag{3.6}$$

La réduction de la dimension des données consiste à supprimer les composantes dont les inerties sont les plus faibles. La matrice $W^\star$ est tronquée et devient $W_q^\star = [\mathbf{w_1^\star} \dots \mathbf{w_q^\star}]$ avec $q < n$. Les données réduites s'expriment alors par :

$$\mathbf{y_i^\star} = W^{\star\perp}(\mathbf{y_i} - \bar{\mathbf{y}}) \tag{3.7}$$

L'erreur de réduction peut s'évaluer par le vecteur $\mathbf{e_i} = \mathbf{y_i} - \mathbf{y_i^\star}$ et la quantité d'inertie ou d'information conservé par $\frac{\sum_{k=1}^{q} \lambda_k^\star}{\sum_{k=1}^{n} \lambda_k^\star}$. Notons que toute reduction faite par ACP minimise la quantité $E^\perp E$ où $E = [\mathbf{e_1} \dots \mathbf{e_q}]$.

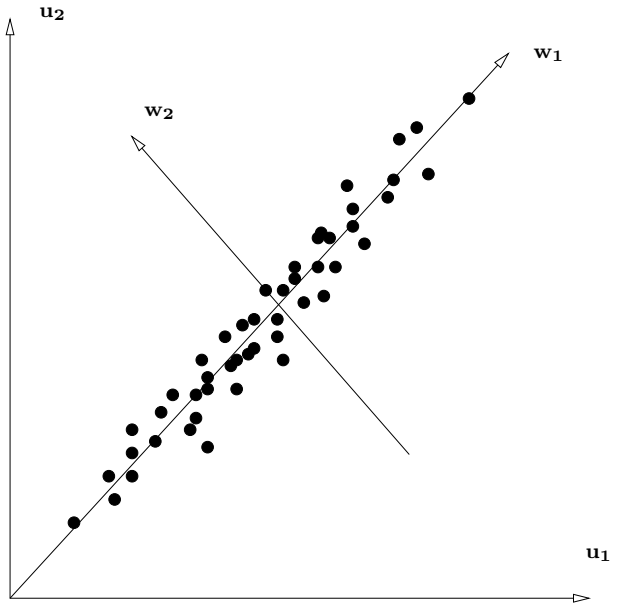


FIGURE 3.1 – Un nuage de point de dimension deux dans un repère canonique initial $(\mathbf{u_1}, \mathbf{u_2})$ avec le repère de l'ACP $(\mathbf{w_1}, \mathbf{w_2})$. La composante principale correspond à projection sur l'axe $\mathbf{w_1}$.

La figure 3.1 présente un exemple d'ACP en dimension deux. Remarquons que nous nous sommes placé ici dans un espace euclidien. Si maintenant l'espace des attributs est muni d'une métrique pondérée par la matrice $M = TT^\perp$ (voir équation 2.63 page 32), les résultats se généralisent facilement. La matrice de covariance munie de cette métrique devient $TXP(TX)^\perp$ et le changement de coordonnées s'effectue avec $\mathbf{y_i} = W^\perp T(\mathbf{x_i} - \bar{\mathbf{x}})$.

3.2 La classification, application à la classification supervisée

Réaliser la classification d'une base d'observations (que nous appelerons base d'apprentissage par rapport à la problématique de la reconnaissance), consiste à trouver les frontières des classes qui la compose. Le terme de classification regroupe en fait trois cas de figure qui impliquent des stratégies différentes. Soit les observations de la base sont étiquetées et il s'agit d'une classification *supervisée*, soit seulement une partie des observations est étiquetée et il s'agit d'une classification *semi-supervisée*, soit enfin aucune des observations n'est étiquetée et il s'agit d'une classification *automatique*. Il est évident que la complexité de la tâche est croissante avec le nombre de données non étiquetées. Dans le cas de la reconnaissance de caractères avec une base fiable, la classification est supervisée ce qui nous le verrons, simplifie considérablement le problème.

Nous avons placé volontairement nos travaux de reconnaissance statistique dans le cadre probabiliste qui possède un formalisme bien établi et dont les convergences sont démontrées. L'objectif est d'appuyer nos résultats sur une bonne compréhension des algorithmes et modélisations mises en jeu. Les approches probabilistes de la classsification reposent sur l'estimation des distributions de chacune des classes de la base d'apprentissage. Deux familles de méthodes sont couramment utilisées : les approches *paramétriques* qui présupposent la forme des lois ; et les approches *non paramétriques* qui les estiment sans connaissance *a priori*.

3.2.1 Les approches paramétriques

Nous présentons dans cette section les stratégies de classification basées sur une paramétrisation des lois de distribution. Nous bornerons notre exposé aux mélanges gaussiens multivariés en décrivant les contraintes imposables pour simplifier le modèle. En nous plaçant dans le cadre de notre problème défini précédemment, nous décrirons enfin comment obtenir une classification optimale.

3.2.1.1 Les modèles de mélange

L'objet de la classification paramétrique est de définir un modèle de distribution pour m observations, $\{\mathbf{x_i}\},\ i = 1, \ldots, m$ vecteurs de $\mathbb{R}^n$, réalisations de m vecteurs aléatoires, $\{X_i\}$ indépendants. Connaissant *a priori* ou non les c classes qui les différencient, il est au moins naturel de supposer que ces observations ont été générées à partir de c distributions homogènes, c'est-à-dire concentrées autour de leur valeur centrale et se chevauchant peu. Alors chaque classe se caractérise par un jeu de paramètres de lois $\phi_k,\ 1 \leq k \leq c$ inconnu et par une proportion π_k du mélange (avec $\sum_{k=1}^{c} \pi_k = 1$). La densité de probabilité en un point $\mathbf{x_i}$ de $\mathbb{R}^n$ est donnée par :

$$p(\mathbf{x_i}|\theta) = \sum_{k=1}^{c} \pi_k f(\mathbf{x_i}|\phi_k) \tag{3.8}$$

où θ est le vecteur de tous les paramètres inconnus du mélange. Dans le cas général considéré, $\theta = (\pi_1, \ldots, \pi_c, \phi_1, \ldots, \phi_c)$ et si les X_i sont considérés indépendants[1], sa vraisemblance s'exprime par :

$$p(\mathbf{x}|\theta) = p(\mathbf{x_1}, \ldots, \mathbf{x_m}|\theta) = \prod_{i=1}^{m} p(\mathbf{x_i}|\theta) = \prod_{i=1}^{m} \sum_{k=1}^{c} \pi_k f(\mathbf{x_i}|\phi_k) \tag{3.9}$$

3.2.1.2 Les mélanges gaussiens

Quand les données sont à valeurs réelles et en l'absence de toute connaissance sur leur distribution, il est courant de supposer que chacune des classes suit une distribution gaussienne multivariée. Ainsi la densité en un point $\mathbf{y} \in \mathbb{R}^n$ conditionnellement à la classe k s'exprime :

$$f(\mathbf{y}|\phi_k) = (2\pi)^{-\frac{n}{2}} |\Sigma_k|^{-1/2} e^{\left(-\frac{1}{2}(\mathbf{y}-\mu_\mathbf{k})^\perp \Sigma_k^{-1}(\mathbf{y}-\mu_\mathbf{k})\right)} \tag{3.10}$$

où Σ_k est la matrice de covariance de la distribution, $|\Sigma_k|$ son déterminant et $\mu_\mathbf{k}$ le vecteur moyenne.

Dans le cas d'un mélange gaussien, le vecteur des paramètres inconnus du modèle est :

$$\theta = (\pi_1, \ldots, \pi_c, \mu_\mathbf{1}, \ldots, \mu_\mathbf{c}, \Sigma_1, \ldots, \Sigma_c) \tag{3.11}$$

[1] hypothèse naïve ([Ris01]).

Ceci caractérise un problème général avec peu d'information *a priori* si ce n'est la forme de la distribution. La matrice de covariance, par exemple, est une matrice symétrique définie positive de dimension $n \times n$. Son évaluation nécessite donc d'estimer $\frac{(n+1)n}{2}$ coefficients (partie triangulaire supérieure ou inférieure), ce qui peut se révéler délicat et couteux. Une autre approche possible se base sur des modèles plus contraints en établissant quelques hypothèses sur la matrice de covariance. CELEUX ET GOVAERT ([CG95]) proposent une série de modèles gaussiens *parcimonieux* reposant sur une décomposition de cette matrice.

Modèles gaussiens parcimonieux L'idée est de diagonaliser la matrice de covariance sous la forme :

$$\Sigma_k = \lambda_k D_k A_k D_k' \tag{3.12}$$

où $\lambda_k = |\Sigma_k|^{\frac{1}{n}}$, appelé *volume* de la classe k, indique la dispersion de la classe dans $\mathbb{R}^n$. La matrice D_k est la matrice de rotation des vecteurs propres ($D_k D_k^{\perp} = I_n$) et est appelée *orientation*. Et enfin $A_k = \mathrm{diag}(a_{k_1}, \ldots, a_{k_n})$ est la matrice des valeurs propres normalisées (de déterminant égal à 1) et est appelée *forme*.

Voici trois propositions intéressantes dont les démonstrations se trouvent dans la thèse de DANG ([Dan98]) :

Proposition 1 *Pour une distribution gaussienne dans $\mathbb{R}^n$, $\mathcal{N}(\mu; \Sigma = \lambda DAD')$, le lieu des points $\mathbf{y} \in \mathbb{R}^n$ ayant une densité donnée est tel que la distance de Mahalanobis de $\mathbf{y}$ à μ est constante :*

$$d_M^2(\mathbf{y}, \mu) = (\mathbf{y} - \mu)^{\perp} \Sigma^{-1} (\mathbf{y} - \mu) = \Delta \tag{3.13}$$

où Δ est une constante positive qui dépend du niveau de densité choisi.

Proposition 2 *Si Y est un vecteur aléatoire de $\mathbb{R}^n$ de densité de probabilité $\mathcal{N}(\mu, \Sigma)$, la probabilité que Y se trouve à une distance de Mahalanobis de μ inférieure ou égale à Δ est donné par :*

$$p_\Delta(Y) = Pr(d_M^2(\mathbf{y}, \mu) \leq \Delta) = 1 - e^{-\frac{\Delta}{2}} \tag{3.14}$$

Et enfin :

Proposition 3 *Si Y est un vecteur aléatoire de $\mathbb{R}^n$ de densité de probabilité $\mathcal{N}(\mu, \Sigma)$, toute observation $\mathbf{y}$ de Y est une composition linéaire d'une observation d'un vecteur aléatoire $\mathbf{u}$ de $\mathbb{R}^n$ et de densité de probabilité $\mathcal{N}(\mathbf{0}, I)$ (où $\mathbf{0}$ est le vecteur nul et I la matrice identité) suivant la relation :*

$$\mathbf{y} = \Sigma^{\frac{1}{2}} \mathbf{u} + \mu \tag{3.15}$$

La dernière proposition est souvent utilisée pour la simulation d'un vecteur gaussien à partir d'un vecteur aléatoire gaussien standard d'un ensemble de points. En dimension deux, avec les notations suivantes :

$$A = \begin{bmatrix} a & 0 \\ 0 & \frac{1}{a} \end{bmatrix} \quad D = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{3.16}$$

Le lieu des points de probabilité p_Δ est une ellipse centrée en μ et dont le grand et le petit axe ont pour longueurs respectives $\sqrt{\Delta\lambda a}$ et $\sqrt{\Delta\frac{\lambda}{a}}$. Cette ellipse est appelée *ellipse de dispertion de niveau p_Δ*. La figure 3.2 page suivante présente l'ellipse de dispersion $p_1 \approx 0,39$.

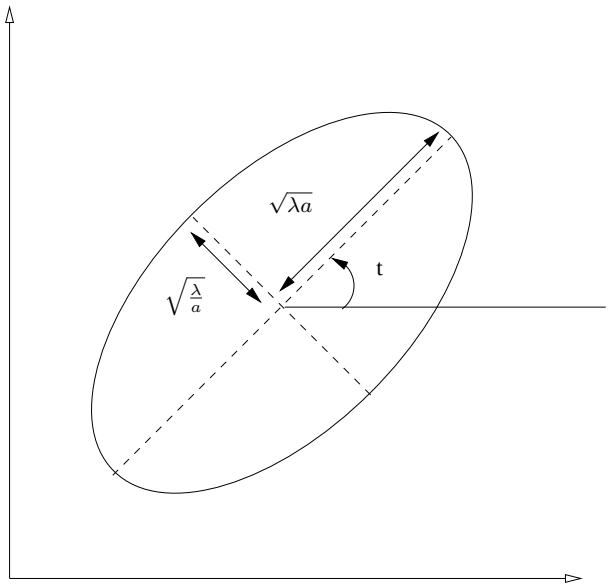


FIGURE 3.2 – Ellipse de dispertion de probabilité $p_1 \approx 0,39$

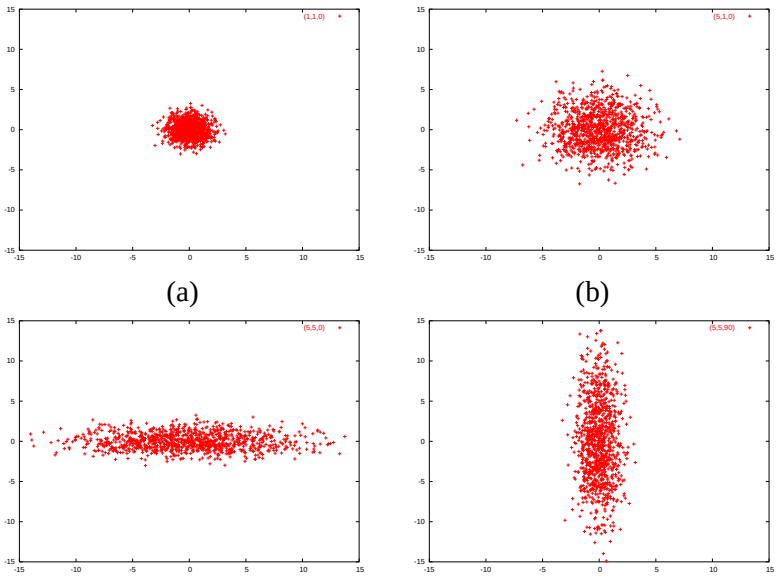


FIGURE 3.3 – Simulations de 1 000 points gaussiens en dimension deux suivant les paramètres (a)$\lambda = 1, a = 1, \theta = 0$, (b) $\lambda = 5, a = 1, \theta = 0$, (c) $\lambda = 5, a = 5, \theta = 0$ et (d) $\lambda = 5, a = 5, \theta = \frac{\pi}{2}$

La figure 3.3 donne quatre résultats de simulation de points gaussiens en dimension deux avec différentes valeurs de λ, a et θ.

D'une manière générale, en imposant des contraintes sur les trois paramètres de la décomposition de la matrice de covariance, les distributions gaussiennes peuvent se classer en trois familles :

- famille sphérique : $A_k = I$,
- famille diagonale : l'orientation est parallèle aux axes donc D est une matrice de permutation de la base canonique et Σ est une matrice diagonale.
- famille générale : le volume, la forme et l'orientation sont quelconques.

Modèles de variances Il est possible de faire également des hypothèses sur l'ensemble des matrices de covariances du mélange. Les quatre hypothèses couramment utilisées sont :

- Modèle linéaire sphérique : $\Sigma_1 = \ldots = \Sigma_c = \sigma I$. Les variables sont indépendantes et possèdent toutes, quelque soit la classe, la même variance.
- Modèle linéaire diagonal : $\Sigma_1 = \ldots = \Sigma_c = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_n^2)$. Les variables sont indépendantes et leurs variances sont indépendantes des classes.
- Modèle linéaire général : $\Sigma_1 = \ldots = \Sigma_c$. Les variables ne sont pas indépendantes mais leurs corrélations et variances sont indépendantes des classes.
- Modèle quadratique : aucune contrainte n'est imposée sur les Σ_k, les variables sont corrélées et les corrélations ainsi que les variances peuvent dépendre des classes.

3.2.1.3 Estimation des paramètres de mélange

Cadre général Revenons au problème général de la classification paramétrique comme la définition du modèle de distribution pour m observations, $\{\mathbf{x_i}\},\ i = 1, \ldots, m$ vecteurs de $\mathbb{R}^n$, réalisations de m vecteurs aléatoires, $\{X_i\}$ indépendants. Nous avons vu que la vraisemblance des observations était la probabilité qu'elles se réalisent conditionnellement aux paramètres θ du mélange (équation 3.9 page 38). Le logarithme de la vraisemblance noté $L(\theta)$ s'exprime donc :

$$L(\theta) = \sum_{i=0}^{m} \log \sum_{k=0}^{c} \pi_k f(\mathbf{x_i}|\phi_k) \tag{3.17}$$

Le principe du maximum de vraisemblance est d'estimer θ par le jeu de paramètres $\hat{\theta}_{MV}$ qui maximise la vraisemblance (ou son $\log$ que nous confondrons par commodité) :

$$\hat{\theta}_{MV} = \arg\max_{\theta} (L(\theta)) \tag{3.18}$$

$\hat{\theta}_{MV}$ est un estimateur asymptotiquement non biaisé[2], efficace[3] et gaussien. Il est solution de l'équation :

$$\frac{\partial L(\theta)}{\partial \theta} = 0 \tag{3.19}$$

$\hat{\theta}_{MV}$ doit être un point critique de la vraisemblance[4] et un minimum relatif[5]. Pour les mélanges, il n'existe pas de solution analytique au cas général et il faut recourir à des méthodes itératives (l'algorithme *Expectation Maximization* et ses dérivés que nous ne présenterons pas ici mais pour lesquels il existe de nombreux travaux dont, entre autres, les thèses de SAINT-JEAN ([SJ01]) ou de DANG ([Dan98])).

Une autre approche est l'estimation bayésienne. Il faut alors utiliser une fonction de coût $l(\hat{\theta}|\theta)$ et une distribution *a priori* des paramètres $p(\theta)$. L'estimateur bayésien $\hat{\theta}_B$ correspond alors au jeu de paramètres qui minimise la fonction de risque définie par :

$$\mathcal{R}(\hat{\theta}|\mathbf{x}) = \int_{\theta} l(\hat{\theta}|\theta) p(\theta|\mathbf{x}) d\theta \tag{3.20}$$

où $p(\theta|\mathbf{x})$ est la probabilité de θ conditionnellement à $\mathbf{x}$ dite probabilité *a posteriori*. Elle s'exprime en fonction de la probabilité *a priori* et de la vraisemblance grâce à la relation de Bayes :

$$p(\theta|\mathbf{x}) = \frac{p(\mathbf{x}|\theta)p(\theta)}{\int_{\theta} p(\theta)p(\mathbf{x}|\theta)d\theta} \tag{3.21}$$

Le choix de la fonction de coût détermine le type de l'estimateur bayésien. Typiquement :

[2] $\lim_{m \to +\infty}(E(\hat{\theta}) - \theta) = 0$

[3] il minimise l'erreur quadratique moyenne $E\left(|\hat{\theta} - \theta|^2\right)$

[4] un point critique annule les dérivées partielles par rapport à chacune des composantes de θ

[5] la matrice hessienne en ce point doit avoir ses valeurs propres strictement négatives

– $\hat{\theta}_B$ est le *Maximum A Posteriori* (MAP) si le coût est du type $0-1$:

$$l(\hat{\theta}|\theta) = \begin{cases} 1 \text{ si } \hat{\theta} \neq \theta \\ 0 \text{ sinon} \end{cases} \tag{3.22}$$

alors :

$$\hat{\theta}_B = \arg\max_{\theta} p(\theta|\mathbf{x}) \tag{3.23}$$

– $\hat{\theta}_B$ est le *Mode Postérieur Maximum* (MPM) si le coût se décompose sur chaque composante de θ de la manière suivante :

$$l(\hat{\theta}|\theta) = \sum_{\hat{\theta}_t \neq \theta_t} 1 \tag{3.24}$$

alors :

$$\hat{\theta}_B = \{\hat{\theta}_t = \arg\max_{\theta_t} p(\theta_t|\mathbf{x})\} \tag{3.25}$$

Dans le cas général, la distribution *a posteriori* $p(\theta|\mathbf{x})$ est difficile à estimer et il faut recourir à des techniques d'échantillonnage du type MCMC (*Markov Chain Monte Carlo*).

Cas particulier de la classification supervisée Dans la cadre de la classification supervisée, la base d'apprentissage est étiquetée et chaque observation est attachée à une classe et une seule par la fonction d'étiquetage ϵ. Les probabilités *a priori* des classes peuvent s'estimer facilement par leur fréquence d'apparition dans la base :

$$\pi_k = \frac{m_k}{m} \tag{3.26}$$

où m_k désignera le nombre d'éléments appartenant à la classe $\mathcal{C}_k$ d'étiquette ω_k. Cependant lorsque le nombre d'éléments par classe est très faible, cette estimation devient relativement imprécise et peut être remplacée par un estimateur Laplacien du type :

$$\pi_k = \frac{m_k + M/c}{m + M} \tag{3.27}$$

où M est une constante. Remarquons qu'avec cette formulation, la somme des π_k n'est plus égale à 1 ce qui induit l'existence de classes de rejet. Dans la suite, nous considérerons les π_k comme constants et connus, leurs valeurs seront explicitées dans le paragraphe 3.3 page 47. Le problème de classification revient alors à l'estimation des $\{\phi_k\}$ paramètres des lois de distribution de chacune des classes. Dans le cadre d'une estimation paramétrique gaussienne, $\phi_k = \mu, \Sigma_k$ et l'utilisation de l'estimateur du maximum de vraisemblance donne :

$$\hat{\mu}_k = \frac{1}{m_k} \sum_{i=1}^{m_k} \mathbf{x}_i^{\mathcal{C}_k} \tag{3.28}$$

$$\hat{\Sigma_k} = \frac{1}{m_k - 1} \sum_{i=1}^{m_k} (\mathbf{x}_i^{\mathcal{C}_k} - \hat{\mu}_k)(\mathbf{x}_i^{\mathcal{C}_k} - \hat{\mu}_k)^{\perp} \tag{3.29}$$

où $\mathbf{x}_i^{\mathcal{C}_k}$ désigne la i$^{\text{e}}$ observation de la classe $\mathcal{C}_k$.

Rappelons que ces estimateurs sont asymptotiquement non-biaisés et efficaces, ils nécessitent donc un nombre suffisant d'éléments par classes. L'utilisation d'un estimateur bayésien de type MAP (à l'aide d'hypothèses sur les distributions *a priori* des paramètres) donne des expressions plus complexes qu'il est difficile d'utiliser sans hypothèse sur les matrices de covariance. Elles sont utiles pour des bases peu fournies mais convergent vers les estimations précédentes pour un nombre d'éléments par classe suffisant ([CMK02]). Nous bornerons notre étude des modèles paramétrés à ces estimations mais beaucoup de

travaux récents apportent des réponses aux problématiques de la robustesse aux données manquantes ou aberrantes (voir [SJ01]).

A partir de cette paramétrisation et de la règle de Bayes, les lois *a posteriori* d'observation des étiquettes conditionnellement à une observation $\mathbf{y}$ s'expriment simplement par :

$$\hat{p}(\omega_k|\mathbf{y}) = \frac{\pi_k f(\mathbf{y}|\omega_k)}{\underbrace{\sum_j \pi_j \hat{f}(\mathbf{y}|\omega_j)}_{\hat{f}(\mathbf{y})}} = \frac{\pi_k}{(2\pi)^{\frac{n}{2}}\sqrt{|\hat{\Sigma}_k|}} \frac{e^{\left(-\frac{1}{2}(\mathbf{y}-\hat{\mu}_\mathbf{k})^\perp \hat{\Sigma}_k^{-1}(\mathbf{y}-\hat{\mu}_\mathbf{k})\right)}}{\hat{f}(\mathbf{y})} \tag{3.30}$$

L'estimation de $\hat{f(y)}$ n'est pas toujours réalisable. En revanche elle possède la propriété intéressante de ne pas dépendre des ω_k et peut être considérée comme constante dans le processus de décision.

3.2.2 Les approches non-paramétriques

L'objectif principal des approches non-paramétriques est d'estimer les densités de probabilité des classes sans effectuer, ou presque, d'hypothèse sur leurs lois.

3.2.2.1 Fenêtre de Parzen

Une approche classique consiste à estimer la densité de probabilité sur un voisinage du vecteur considéré. Considérons une région $\mathcal{R}$ de $\mathbb{R}^n$. La probabilité qu'une observation $\mathbf{y}$ du vecteur aléatoire X de densité de probabilité $f(\mathbf{y})$ tombe dans cette région s'exprime :

$$P = \int_\mathcal{R} f(\mathbf{y}) d\mathbf{y} \tag{3.31}$$

Si $\mathcal{R}$ est suffisamment petite, f est constante sur $\mathcal{R}$ et l'équation précédente devient :

$$P = V.f(\mathbf{y}) \tag{3.32}$$

où V désigne le volume de $\mathcal{R}$. Considérons maintenant la variable aléatoire R définie par la probabilité que r observations de X tombent dans $\mathcal{R}$. Cette variable aléatoire suit une loi binômiale $\mathcal{B}(P, m)$ où P représente bien la réalisation d'une observation et m est le nombre total d'observations de X. Pour une loi binômiale, P est classiquement estimée par :

$$\hat{P} = \frac{r}{m} \tag{3.33}$$

Ainsi, en remplaçant P par sa valeur dans l'équation 3.32, la densité de probabilité de X sur $\mathcal{R}$ s'estime par :

$$\hat{f}(\mathbf{y}) = \frac{r/m}{V} \tag{3.34}$$

La convergence de $\hat{f}(\mathbf{y})$ vers $f(\mathbf{y})$ est vraie dans les conditions suivantes :

$$\begin{array}{l} \lim_{m\to+\infty} V = 0 \\ \lim_{m\to+\infty} r = +\infty \\ \lim_{m\to+\infty} \frac{r}{m} = 0 \end{array} \tag{3.35}$$

PARZEN ([Par62]) généralise ce résultat en exprimant le nombre d'observations qui tombent dans $\mathcal{R}$ à l'aide d'une fonction noyau $\delta : \mathbb{R} \to \mathbb{R}^+$, appelée *fenêtre de Parzen*. Dans le cas d'un voisinage hypercubique de dimension d et de côté h centré en $\mathbf{y}$, la fenêtre sera de la forme :

$$\delta(x) = \begin{cases} 1 \text{ si } x^j < \frac{1}{2}\ \forall j \in \{1, \ldots, d\} \\ 0\ sinon \end{cases} \tag{3.36}$$

Uniforme	$\delta(x) = \begin{cases} \frac{1}{2} \text{ si } \lvert x\rvert \leq 1 \\ 0 \text{ sinon} \end{cases}$
Linéaire	$\delta(x) = \begin{cases} 1 - \lvert x\rvert \text{ si } \lvert x\rvert \leq 1 \\ 0 \text{ sinon} \end{cases}$
Gaussien	$\delta(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$
Exponentiel	$\delta(x) = \frac{1}{2} e^{-\lvert x\rvert}$
Epanechnikov	$\delta(x) = \begin{cases} \frac{3}{4} \frac{(1-\frac{x^2}{5})}{\sqrt{5}} \text{ si } \lvert x\rvert \leq \sqrt{5} \\ 0 \text{ sinon} \end{cases}$

TABLE 3.1 – Fenêtres de Parzen.

Le volume de $\mathcal{R}$ vaut $V = h^d$ et $\delta\left(\frac{\|\mathbf{y}-\mathbf{x_i}\|}{h}\right) = 1$ si $\mathbf{x_i}$ tombe dans $\mathcal{R}$. L'équation 3.34 page précédente peut alors s'écrire :

$$\hat{f}(\mathbf{y}) = \frac{1}{m} \sum_{i=1}^{m} \frac{\delta\left(\frac{\|\mathbf{y}-\mathbf{x_i}\|}{h}\right)}{h^d} \tag{3.37}$$

$\delta(x)$ est une fenêtre de lissage. Il est possible de lui affecter d'autres valeurs à condition que $\hat{f}$ reste une densité ce qui implique que les conditions suivantes soient respectées :

$$\delta(x) > 0 \tag{3.38}$$

et

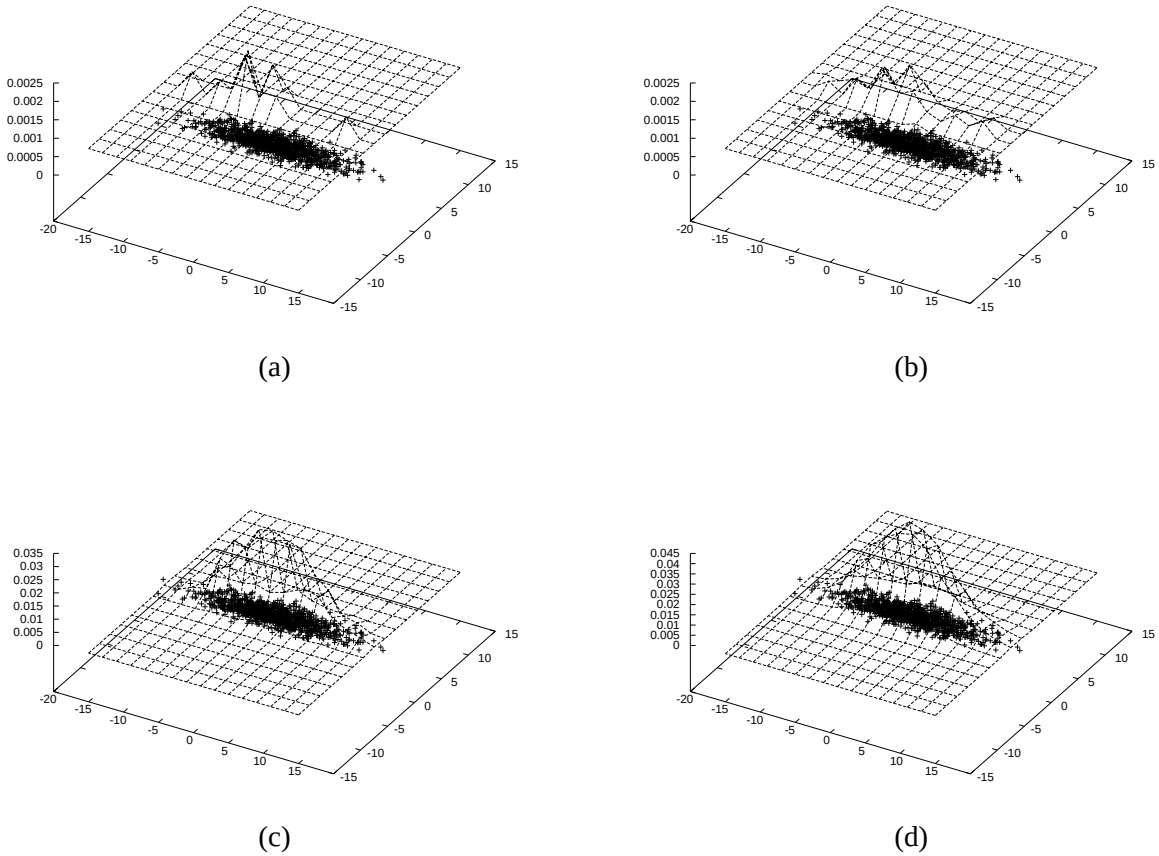
$$\int \delta(\|\mathbf{y}\|) d\mathbf{y} = V \tag{3.39}$$

Dans l'equation 3.37 la fenêtre de lissage est pondérée par h appelée *largeur de bande*. Plus h est élevée, plus grand est le nombre d'observations qui tombent dans $\mathcal{R}$. La largeur de bande est donc à relier à la forme du voisinage choisi. Si h est trop grand alors $\hat{f}$ présentera un caractère extrêmement lissé mais à l'inverse si h est trop petit $\hat{f}$ sera relativement bruitée tendant vers une somme d'impulsions autour de chaque observation.

La table 3.1 donne quelques fenêtres courantes. La figure 3.4 page suivante présente les estimations de la probabilité d'observation dans un cas de dimension deux avec les fenêtres uniformes et gaussiennes suivant deux valeurs de h. Le choix de h est un problème complexe et plusieurs solutions ont été proposées ([MA04]). Une solution simple et répandue est de prendre $h = m^{-\alpha/2}$ avec $0 < \alpha < 1$.

Dans le cadre de la classification supervisée, c'est la loi de probabilité de X conditionnellement à une étiquette de classe ω_k qui est estimée par :

$$\hat{f}(\mathbf{y}|\omega_k) = \frac{1}{m_k} \sum_{i=1}^{m_k} \frac{\delta\left(\frac{\left\|\mathbf{y}-\mathbf{x}_i^{\mathcal{C}_k}\right\|}{h}\right)}{V} \tag{3.40}$$



(a) (b)

(c) (d)

FIGURE 3.4 – Probabilités d'observations estimées avec les fenêtres de Parzen. Les observations sont les points noirs du plan (x, y), la surface représente la probabilité suivant l'axe z. Pour (a) et (b) $h = 0, 1$ et les fenêtres utilisées sont repectivement la fenêtre uniforme et gaussienne. Pour (c) et (d) h vaut 1 et les fenêtres sont les mêmes qu'au dessus.

où $\mathbf{x}_i^{\mathcal{C}_k}$ désigne la i$^{\text{e}}$ observation de la classe $\mathcal{C}_k$. La loi *a posteriori* d'observation d'une étiquette conditionnellement à une observation s'obtient avec la règle de Bayes :

$$\hat{p}(\omega_k|\mathbf{y}) = \underbrace{\frac{\pi_k \hat{f}(\mathbf{y}|\omega_k)}{\sum_j \pi_j \hat{f}(\mathbf{y}|\omega_j)}}_{\hat{f}(\mathbf{y})} = \frac{m \times \pi_k}{m_k} \frac{\sum_{i=1}^{m_k} \delta\left(\frac{\left\|\mathbf{y}-\mathbf{x}_i^{\mathcal{C}_k}\right\|}{h}\right)}{\sum_{i=1}^{m} \delta\left(\frac{\|\mathbf{y}-\mathbf{x}_i\|}{h}\right)} \tag{3.41}$$

3.2.2.2 La règle des k Plus Proches Voisins (k-PPV)

La règle des k-PPV et celle du PPV explicitées dans le paragraphe suivant ont été formalisées dans les années 50 par FIX ET HODGES [CH67]. La règle du k-PPV consiste à baser le choix du voisinage $\mathcal{R}$ sur le nombre d'observations qu'il contient fixé à k. C'est le volume V_k qui est variable. L'estimateur de la densité de probabilité s'exprime alors de la façon suivante :

$$\hat{f}(\mathbf{y}) = \frac{k}{m \times V_k} \tag{3.42}$$

Cette estimation n'est valable que dans les conditions déjà définies avec l'estimateur de Parzen, c'est-à-dire :

$$\begin{array}{l} \lim_{m\to+\infty} V_k = 0 \\ \lim_{m\to+\infty} \frac{k}{V_k} = 0 \end{array} \tag{3.43}$$

Dans le cadre de la classification supervisée, la loi de probabilité conditionnelle peut être approximée par :

$$\hat{f}(\mathbf{y}|\omega_i) = \frac{k_i}{m_i \times V_k} \tag{3.44}$$

où k_i est le nombre d'observations contenues dans V_k appartenant à la classe d'étiquette ω_i. La loi *a posteriori* d'observation d'une étiquette conditionnellement à une observation s'obtient avec la règle de Bayes :

$$\hat{p}(\omega_i|\mathbf{y}) = \frac{\pi_i \hat{f}(\mathbf{y}|\omega_i)}{\sum_j \pi_j \hat{f}(\mathbf{y}|\omega_j)} = \frac{\pi_i \frac{k_i}{m_i \times V_k}}{\hat{f}(\mathbf{y})} = \frac{\pi_i \times m}{m_i} \frac{k_i}{k} \tag{3.45}$$

Le choix de k est lié à la qualité de la base d'apprentissage. Prendre k élevé permet d'exploiter au mieux les propriétés locales mais nécessite un grand nombre d'échantillons pour contraindre le volume du voisinage à rester petit. Bien souvent k est choisi comme la racine carrée du nombre moyen d'éléments par classe soit $\sqrt{\frac{m}{c}}$.

3.2.2.3 La règle du Plus Proche Voisin (PPV)

Il s'agit d'un cas particulier de la règle des k-PPV où $k = 1$:

$$\hat{f}(\mathbf{y}) = \frac{1}{m \times V} \tag{3.46}$$

et la loi *a posteriori* s'exprime en fonction de l'observation voisine $\mathbf{x}_v$ de $\mathbf{y}$:

$$\hat{p}(\omega_i|\mathbf{y}) = \begin{cases} \frac{\pi_i \times m}{m_i} \text{ si } \epsilon(\mathbf{x}_v) = \omega_i \\ 0 \text{ sinon} \end{cases} \tag{3.47}$$

Le principal inconvénient des approches non-paramétriques basées sur un voisinage est la complexité algorithmique. Elle est d'ordre $o(km)$ dans le cas du k-PPV. Ce problème est particulièrement délicat pour les espaces de représentation d'ordre élevé. Il existe plusieurs méthodes classiques pour réduire cette complexité combinatoire basées sur une simplification des distances, une réduction des échantillons ou une structuration de la base ([BSA92], [MO98], [RP00], [ZS04]).

COVER ET HART ([CH67]) ont montré que les probabilités d'erreurs asymptotiques moyennes des classifieurs par plus proches voisins pouvaient être encadrées par l'erreur bayésienne (Pr_{bayes}, voir définition en annexe B page 167) de la façon suivante :

$$Pr_{bayes} \leq Pr_{kppv} \leq Pr_{(k-1)ppv} \leq \ldots \leq Pr_{ppv} \leq Pr_{bayes}\left(2 - \frac{c}{c-1} Pr_{bayes}\right) \tag{3.48}$$

où P_{kppv} est l'erreur asymptotique moyenne du k-PPV :

$$P_{kppv} = 1 - \sum_{i=1}^{c} \left(\int_{C_i} \hat{p}(\omega_i|\mathbf{y}) \hat{p}(\omega_i) d\mathbf{y} \right) \tag{3.49}$$

avec C_i la région de $\mathbb{R}^n$ où $\epsilon(\mathbf{y})$ maximise asymptotiquement la probabilité *a posteriori* (c'est-à-dire sous la condition $m \to +\infty$).

3.3 Stratégie de décision

Les différentes approches de classification permettent, dans le cadre probabiliste, d'estimer les fonctions de densité des observations de la base d'apprentissage relativement aux classes. Dans le cadre de la classification supervisée, les lois de probabilité des classes sont constantes et estimées à l'aide de leur fréquence d'apparition dans la base. Les lois de probabilité *a posteriori* s'expriment alors facilement en fonction des lois *a priori*. La question de la reconnaissance consiste ensuite à attribuer à une nouvelle observation $\mathbf{y}$ une étiquette de classe par la fonction d'étiquetage ϵ. Si les densités *a priori* des classes et les vraisemblances sont connues, la meilleure décision est la décision bayésienne.

Dans un espace de caractéristiques fixé avec une estimation de la loi *a posteriori*, la reconnaissance se fait à partir de l'estimateur bayésien pour s'assurer de la probabilité d'erreur la plus faible. Pour cela nous avons vu précédemment qu'il fallait se fixer une fonction de coût. Le choix de la fonction de coût détermine la stratégie de reconnaissance adoptée. Comme nous voulons pouvoir adapter la reconnaissance à des bases générales pour lesquelles aucune garantie n'est donnée sur l'exhaustivité de l'apprentissage, il nous faut prévoir la possibilité que $\mathbf{y}$ ne puisse pas être étiqueté. Les raisons d'une non-reconnaissance sont doubles et correspondent, suivant le modèle développé par DUBUISSON ([Dub01]), à deux *rejets* du système de décision :

- le rejet dit de *distance* : une observation sera rejetée en distance si le système de décision juge qu'il n'appartient à aucune des classes apprises ;
- le rejet dit d'*ambiguïté* : une observation sera rejetée en ambiguïté si le système de décision juge qu'il est succeptible d'appartenir à plusieurs classes connues avec la même confiance.

DUBUISSON propose de modéliser le rejet en distance par une nouvelle classe d'étiquette ω_0 dont les lois de probabilité ne sont pas connues. Une fonction de coût possible est :

$$l(\epsilon(\mathbf{y}),\omega_j) = \begin{cases} 0 \text{ si } \epsilon(\mathbf{y}) = \omega_j,\ j = 0,\ldots,c \\ 1 \text{ si } \epsilon(\mathbf{y}) \neq \omega_j,\ j = 0,\ldots,c \\ a \text{ si } \mathbf{y} \text{ est rejeté en ambiguïté, par convention : } \epsilon(\mathbf{y}) = \omega_{-1} \end{cases} \tag{3.50}$$

Et le risque bayésien devient :

$$R(\epsilon(\mathbf{y}),\mathbf{y}) = \sum_{j=0}^{c} l(\epsilon(\mathbf{y}),\omega_j)p(\omega_j|\mathbf{y}) = \begin{cases} a \text{ si } \mathbf{y} \text{ si } \epsilon(\mathbf{y}) = \omega_{-1} \\ 1 - p(\omega_j|\mathbf{y}) \text{ si } \epsilon(\mathbf{y}) = \omega_j,\ j = 0,\ldots,c \end{cases} \tag{3.51}$$

La minimisation du risque bayésien donne la règle de décision :

- $\epsilon(\mathbf{y}) = \omega_i$ si :

$$p(\omega_i|\mathbf{y}) = \max_{j=0,\ldots,c}(p(\omega_j|\mathbf{y})) \text{ et } p(\omega_i|\mathbf{y}) \geq 1 - a \tag{3.52}$$

- $\mathbf{y}$ est rejeté en ambiguïté si :

$$p(\omega_i|\mathbf{y}) = \max_{j=0,\ldots,c}(p(\omega_j|\mathbf{y})) \text{ et } p(\omega_i|\mathbf{y}) \leq 1 - a \tag{3.53}$$

Si les $p(\omega_j|\mathbf{y}),\ j = 1,\ldots,c$, ont été estimés directement, le calcul de la loi de probabilité *a posteriori* de la classe ω_0 se fait par :

$$p(\omega_0|\mathbf{y}) = 1 - \sum_{j=1}^{c} p(\omega_j|\mathbf{y}) \tag{3.54}$$

En effet, l'existence de la classe de rejet en distance ne permet pas d'exprimer les probabilités *a priori* des autres classes simplement de façon fréquentielle comme dans l'équation 3.26 page 42. Il faut soit utiliser l'expression de l'équation 3.27 page 42 et postuler que la seule classe inconnue est ω_0, soit, comme DUBUISSON, utiliser les estimations fréquentielles et considérer l'imprécision dans le rejet.

Alors, à partir des $\pi_k = m_i/m$, les probabilités *a priori* peuvent s'écrire en fonction de $p(\omega_0)$ de la manière suivante :

$$p(\omega_k) = (1 - p(\omega_0))\pi_k \ , \ i = 1, \ldots, c \tag{3.55}$$

et la somme des $p(\omega_k)$ de 0 à c est égale à 1. Avec cette approximation, les π_k des équations 3.30 page 43,
3.41 page 45, 3.45 page 46 et 3.47 page 46 se remplacent par les $p(\omega_k)$ de l'équation précédente. Comme nous ne pouvons privilégier aucune région de l'espace pour l'apparition d'une nouvelle classe, il est naturel de considérer que la densité de probalité d'une observation conditionnellement à ω_0 est uniforme sur un volume V de l'espace suffisamment grand pour contenir toutes les observations possibles. La loi *a priori* d'une observation conditionnellement à ω_0 s'écrit alors :

$$f(\mathbf{y}|\omega_0) = \frac{1}{V}, \ \forall \mathbf{y} \in \mathbb{R}^n \tag{3.56}$$

Le choix de V peut être délicat, il est préférable d'ajouter une autre condition qui est que la probabilité d'apparition d'une nouvelle classe est *a priori* proportionelle à ce volume V. Donc :

$$p(\omega_0) = \gamma V \tag{3.57}$$

ce qui implique :

$$p(\omega_0) f(\mathbf{y}|\omega_0) = \gamma \tag{3.58}$$

et par la règle de Bayes :

$$p(\omega_0|\mathbf{y}) = \frac{\gamma}{\sum_{j=1}^{c} p(\omega_j) f(\mathbf{y}|\omega_j) + \gamma} \tag{3.59}$$

Si nous notons $g(\mathbf{y})$ la densité de probabilité de $\mathbf{y}$ dans le cas où il n'y a pas de rejet en distance alors :

$$\sum_{j=1}^{c} p(\omega_j) f(\mathbf{y}|\omega_j) = (1 - p(\omega_0)) \underbrace{\sum_{j=1}^{c} \pi_j f(\mathbf{y}|\omega_j)}_{g(\mathbf{y})} = (1 - p(\omega_0)) g(\mathbf{y}) \tag{3.60}$$

Et l'équation 3.59 s'écrit :

$$p(\omega_0|\mathbf{y}) = \frac{\gamma}{(1 - p(\omega_0)) g(\mathbf{y}) + \gamma} \tag{3.61}$$

Ce qui permet d'exprimer γ en fonction de $p(\omega_0)$ avec l'équation 3.54 page précédente :

$$\gamma = \frac{g(y)(1 - p(\omega_0))(1 - \sum_{j=1}^{c} p(\omega_j|\mathbf{y}))}{\sum_{j=1}^{c} p(\omega_j|\mathbf{y})} \tag{3.62}$$

Le rejet en distance se fait classiquement par un seuil imposé à la densité de probabilité de l'observation par rapport aux classes non nulles ([DM93]). L'observation $\mathbf{y}$ est rejetée en distance si :

$$g(\mathbf{y}) \leq s_d \tag{3.63}$$

En repartant de la condition du rejet en distance de l'équation 3.52 page précédente, il est possible d'exprimer s_d en fonction de a et de $p(\omega_0)$ de la manière suivante :

$$s_d = \frac{\gamma a}{(1 - a)(1 - p(\omega_0))} \tag{3.64}$$

Ainsi en cas d'estimation directe des $p(\omega_j|\mathbf{y}),\ j = 1, \ldots, c$, l'utilisation de l'équation 3.54 page 47 permet de ne faire dépendre la décision que du choix de a et de $p(\omega_0)$. En cas d'estimation approchée (par exemple si $f(\mathbf{y})$ n'est pas estimable), il faudra ajouter une estimation de γ ou de V dans le système de décision.

Nous avons, dans ce chapitre, redéfini deux grandes familles de classifieurs probabilistes utilisables dans la reconnaissance statistique de caractères. La stratégie de décision exposée à la fin de ce chapitre permet, d'une part, la mise en œuvre des classifieurs dont nous allons évaluer les performances dans le chapitre suivant et, d'autre part, la définition de systèmes coopérants sur lesquels nous reviendrons dans le chapitre 7 page 107.

CHAPITRE 4

RÉSULTATS ET DISCUSSION

Sommaire

Dans ce paragraphe nous donnons quelques résultats de classification avec différents couples de descripteurs, classifieurs. L'objectif est de trouver des combinaisons intéressantes pouvant servir de comparaison et pour une coopération avec les approches structurelles présentées dans les deux prochaines parties. Les tests ont été effectués uniquement sur les bases de caractères simples. Avec ce type de descriptions, les mots et les caractères plus complexes n'ont que peu d'intérêt.

4.1 Classifications simples

Voici les résultats obtenus sur les trois bases de caractères GrCor, GrAnc et MNIST (voir Annexe A page 155). Quatre types de classification ont été utilisés avec une décision bayésienne sans rejet :

- Classification paramétrique gaussienne multivariée sans contrainte sur les matrices de covariance (notée ParamGau).
- Classification par fenêtre de Parzen avec un noyau exponentiel (notée ParzExp).
- Classification par la règle des k-PPV avec une distance euclidienne et de Mahalanobis (notées KppvEucl et KppvMaha).

Nous avons comparé les résultats de classification simple de cinq types d'attributs :

- Les descripteurs de Fourier complexes (notés FourCompl).
- Les descripteurs de Fourier basés sur la distance au centre de gravité (notés FourCdG).
- Les descripteurs de Zernike (notés Zer).
- Les descripteurs de Hu (notés Hu).
- Les descripteurs de Flusser (notés Flusser).

Comme il a été souligné en introduction de ce mémoire, seules les tables de caractères peuvent être utilisées avec ce genre de descripteurs et de classifieurs. Avec les bases de mots ou de caractères idéographiques, le vocabulaire est bien trop large et le nombre de descripteurs nécessaire à une classification

correcte serait trop important.

4.1.1 Base GrCor

Compte-tenu de la taille de la base et du nombre de classes, nous avons utilisé une largeur de bande $h = 0,007$ pour les fenêtres de Parzen et un nombre de voisins $k = 19$ pour les règles des k Plus Proches Voisins. La table 4.1 présente les résultats de classification obtenus par validation croisée. 4 000 caractères ont été tirés au hasard dans la base pour former les bases de tests. Ces 4 000 caractères de test ont été divisés en quatre groupes de 1 000 individus. Pour chaque groupe, un taux d'erreur de classification a été calculé par rapport à une base d'apprentissage composée des 3 000 individus des trois autres groupes de test associés au reste de la base. La table donne la moyenne des erreurs sur les quatre groupes ainsi que l'écart type.

Type de Descripteur	**FourCompl**		**FourCdG**		**Zer** (ordres 7 et 10)		**Hu**	**Flusser**
Nombre de moments	10	20	10	20	19	35	7	11
ParamGau								
Erreur Moyenne	13,1%	9,6%	21,5%	19,9%	3,9%	3,1%	69,6%	73,7%
Ecart Type	1,1%	1,5%	1,4%	0,8%	0,7%	0,5%	1,9%	1,9%
ParzExp								
Erreur Moyenne	5,1%	4,0%	24,2%	27,3%	3,5%	3,1%	48,1%	58,1%
Ecart Type	0,1%	0,5%	0,6%	1,5%	1,0%	0,6%	1,0%	1,5%
KppvEucl								
Erreur Moyenne	7,9%	6,5%	12,7%	12,8%	5,8%	4,0%	37,1%	46,5%
Ecart Type	0,1%	0,7%	0,8%	1,9%	0,8%	0,5%	1,3%	1,0%
KppvMaha								
Erreur Moyenne	18,6%	22,5%	36,6%	48,5%	5,0%	4,6%	44,9%	51,4%
Ecart Type	0,7%	1,1%	1,0%	1,4%	0,8%	0,5%	1,3%	1,8%

TABLE 4.1 – Résultats de classification simple avec la base GrCor

Pour cette base, les descripteurs les plus intéressants sont les moments de Fourier complexes et les moments de Zernike. Les moments de Hu fournissent des résultats médiocres et ceux de Flusser sont systématiquement moins performants que ceux de Hu. Ces résultats confirment la supériorité des descripteurs de Zernike et Fourier par rapport aux invariants reposant sur les descriptions statistiques établies dans beaucoup de travaux ([PC99], [Dom03]). Dans le cas présent, cela s'explique par la petite taille des caractères qui comprennent peu de pixels. Les descriptions statistiques sur ce type de formes sont très peu précises. Pour la même raison, les autres descripteurs atteignent des résultats qui varient peu quand le nombre de moments augmente. Enfin nous constatons que les descripteurs de Fourier basés sur la description complexe du contour obtiennent de meilleurs résultats que ceux qui sont basés sur la distance au centre de gravité. Cela vient *a priori* de la perte d'une partie de l'information angulaire dans le deuxième cas.

Du côté des classifieurs, nous notons que l'évaluation paramétrique gaussienne obtient de bons résultats avec les descripteurs de Zernike, alors que pour les moments de Fourier complexes, l'estimation à l'aide d'une fenêtre de Parzen exponentielle semble la plus intéressante. Globalement les quatre classifieurs atteignent des résultats équivalents avec les descripteurs de Zernike alors que les moments de Fourier sont plus sensibles à ce choix.

4.1.2 Base MNIST

Nous avons utilisé, pour cette base, une largeur de bande $h = 0,01$ pour les fenêtres de Parzen et un nombre de voisin $k = 11$ pour les règles des k Plus Proches Voisins. La table 4.2 présente les résultats de classification obtenus suivant le même principe que précédemment (quatre groupes de 1 000 individus tirés aléatoirement).

Type de Descripteur	**FourCompl**		**FourCdG**		**Zer** (ordres 7 et 10)		**Hu**	**Flusser**
Nombre de moments	10	20	10	20	19	35	7	11
ParamGau								
Erreur Moyenne	33,8%	33,2%	30,9%	30,5%	23,7%	21,6%	71,4%	80,5%
Ecart Type	0,7%	1,3%	0,9%	2,0%	0,4%	1,6%	2,5%	2,4%
ParzExp								
Erreur Moyenne	32,6%	30,5%	38,3%	40,9%	24,6%	18,2%	57,1%	62,3%
Ecart Type	1,1%	0,8%	1,2%	2,0%	1,1%	2,0%	1,0%	1,8%
KppvEucl								
Erreur Moyenne	33,0%	31,5%	26,9%	26,0%	23,9%	18,2%	55,8%	61,0%
Ecart Type	0,6%	0,9%	0,6%	1,5%	0,6%	1,2%	1,6%	1,5%
KppvMaha								
Erreur Moyenne	47,1%	56,1%	57,1%	74,5%	24,6%	22,6%	66,6%	65,2%
Ecart Type	0,6%	2,3%	1,8%	1,4%	1,6%	0,8%	2,7%	1,0%

TABLE 4.2 – Résultats de classification simple avec la base MNIST

Les résultats sont moins bons sur cette base. Nous sommes loin des résultats annoncés par LECUN ([LBBH98]) de $2,4\%$ d'erreur avec un 3-PPV et une distance euclidienne pixel à pixel entre les images. Rappelons cependant que ces résultats n'ont été obtenus que par rapport à la base de test des caractères MNIST. Les résultats présentés par LECUN ont été obtenus avec un apprentissage préalable fait sur 60 000 caractères, ce qui explique en partie la différence. Nous n'avons pas essayé de reproduire les conditions de test de LECUN car ceci n'était pas notre propos et nous considérerons donc nos résultats comme référence pour la suite des comparaisons.

Concernant les performances obtenues, les conclusions sont similaires à celles faites sur la base GrCor, à savoir que les descripteurs statistiques atteignent des résultats médiocres et que les descripteurs de Zernike sont les plus performants et ce indépendemment de la classification. Concernant les moments de Fourier, cette fois-ci il n'y a pas de différence flagrante entre les deux types de description. C'est un résultat attendu car, d'une part, les images sont plus petites ici que pour les caractères grecs et ensuite les chiffres présentent des contours relativement simples. Ainsi, l'information contenue dans les contours est peu importante et ne nécessite pas une description complète.

Les classifieurs sont tous à peu près équivalents avec, comme pour la base GrCor, un score relativement médiocre pour le k-PPV utilisant une distance de Mahalanobis. Les descripteurs n'étant pas combinés, les grandeurs représentées par chaque dimension ne sont pas homogènes. L'approximation gaussienne induite par la distance de Mahalanobis n'est pas adaptée à ce cas.

4.1.3 Base GrAnc

La base GrAnc étant plus petite, la largeur de bande utilisée est de $h = 0,005$ pour les fenêtres de Parzen et le nombre de voisin de $k = 7$ pour les règles du plus proche voisin. La table 4.3 page suivante présente les résultats de classification obtenus avec quatre groupes de 200 individus tirés aléatoirement. Le nombre d'éléments par classe étant très inhomogène, nous n'avons testé ni l'évaluation paramétrique gaussienne ni la règle du k-PPV avec une distance de Mahalanobis qui nécessitent un calcul de l'inverse

de la matrice de covariance. Celle-ci n'est, d'une part, pas toujours définie et n'a d'autre part que peu de sens sur les classes avec peu d'éléments.

Type de Descripteur	**FourCompl**		**FourCdG**		**Zer** (ordres 7 et 10)		**Hu**	**Flusser**
Nombre de moments	10	20	10	20	19	35	7	11
ParzExp								
Erreur Moyenne	15,1%	15,3%	85,5%	91,9%	11,8%	27,0%	48,5%	58,0%
Ecart Type	2,0%	2,0%	0,7%	1,2%	4,3%	2,9%	3,2%	3,7%
KppvEucl								
Erreur Moyenne	18,2%	17,5%	24,0%	22,4%	12,7%	8,7%	50,2%	62,5%
Ecart Type	2,5%	1,1%	4,8%	0,6%	0,9%	1,6%	2,1%	4,9%

TABLE 4.3 – Résultats de classification simple avec la base GrAnc

Sur cette base nous constatons également la supériorité des descripteurs de Zernike. La fenêtre de Parzen exponentielle est relativement sensible aux bruits ce qui explique ses performances médiocres sur les moments de Fourier calculés sur la distance au centre de gravité qui sont eux aussi très sensibles aux bruits de contour. Or les contours des caractères de cette base sont beaucoup plus bruités que pour les autres. Cette remarque est valable pour la forme également et c'est ce qui explique pourquoi quand la dimension des descripteurs de Zernike augmente la fenêtre de Parzen obtient des résultats dégradés.

4.2 Combinaison de descripteurs

A l'issue des résultats précédents, nous avons regardé les taux d'erreurs de classification à partir d'une description avec 19 moments de Zernike (ordre 7) et 10 moments de Fourier complexes. Trois combinaisons ont été testées, l'ensemble des moments, après une ACP ne conservant que 10 axes et après une ACP ne conservant que 20 axes.

4.2.1 Base GrCor

Dans ce cas, les tests ont été réalisés sur une base unique de test de 8 400 caractères tirés aléatoirement, et les autres ont servi de base d'apprentissage. La raison de ce choix vient de l'ACP qui utilise, dans notre implémentation, une librairie de calcul matriciel ne supportant pas les matrices de taille trop importante. La table 4.4 présente les résultats obtenus avec $k = 19$ et $h = 0,07$.

Type de Descripteur	**Tous les descripteurs**	**Après ACP**	
Nombre de moments	29	10	20
ParamGau			
Erreur	6,1%	7,1%	3,7%
ParzExp			
Erreur	2,6%	72,9%	83,9%
KppvEucl			
Erreur	3,6%	7,0%	4,0%
KppvMaha			
Erreur	7,9%	10,7%	6,3%

TABLE 4.4 – Combinaison de 10 moments de Fourier complexes et 19 moments de Zernike avec la base GrCor

Nous constatons que la combinaison des descripteurs n'obtient globalement pas de meilleurs résultats que l'utilisation des descripteurs de Zernike seuls. L'estimation par fenêtre de Parzen semble améliorer sensiblement les résultats mais si nous considérons l'écart type obtenu avec les 19 descripteurs de Zernike (table 4.1 page 52) qui était de $1,0\%$, nous voyons que le passage d'un score de $3,5\%$ à $2,6\%$ reste dans la fourchette d'incertitude. L'ACP est intéressante avec 20 axes puisqu'elle garde a peu près les mêmes perfomances qu'avec les 29 descripteurs. Seule la fenêtre de Parzen obtient des résultats très dégradés après l'ACP qui nécéssiterait une réévaluation de la largeur de bande.

4.2.2 Base MNIST

Pour la base MNIST nous avons réutilisé la technique de validation croisée sur quatre groupes de 1 000 individus. Les paramètres sont toujours $h = 0,01$ et $k = 11$.

Type de Descripteur	**Tous les descripteurs**	**Après ACP**	
Nombre de moments	29	10	20
ParamGau			
Erreur	18,2%	27,7%	21,1%
Ecart type	0,9%	0,8%	0,7%
ParzExp			
Erreur	21,2%	88,9%	89,7%
Ecart type	2,1%	0,7%	0,1%
KppvEucl			
Erreur	22,1%	27,1%	22,1%
Ecart type	0,7%	1,4%	1,5%
KppvMaha			
Erreur	22,6%	27,8%	22,8%
Ecart type	1,1%	1,6%	1,5%

TABLE 4.5 – Combinaison de 10 moments de Fourier complexes et 19 moments de Zernike avec la base MNIST

Pour cette base tous les classifieurs atteignent des résultats sensiblement meilleurs avec les 29 descripteurs qu'avec les descripteurs de Zernike ou de Fourier séparément présentés à la table 4.2 page 53. L'amélioration la plus flagrante est celle du k-PPV avec la distance de Mahalanobis qui prend tout son sens ici vu que les attributs ne sont pas homogènes. L'ACP conserve ces performances avec 20 axes ce qui est aussi un résultat intéressant permettant de réduire la complexité du système. Nous constatons enfin toujours la même dégradation des scores pour la fenêtre de Parzen après ACP sans optimisation de la largeur de bande.

La table 4.6 page suivante présente la matrice de confusion pour la reconnaissance sur tous les descripteurs par k-PPV et la table 4.7 page suivante donne la matrice après ACP en conservant 20 axes. Les grandeurs représentent des pourcentages de bonnes reconnaissances comme il est d'usage pour les matrices de confusion. Les pourcentages sont une moyenne sur les quatre groupes et le nombre total cumulé d'éléments inconnus par classe est donné dans la dernière colonne. La dernière ligne donne la pertinence de la classe suivant la formule :

$$P = \begin{cases} \frac{\alpha}{\alpha+\gamma} \text{ si } \alpha + \gamma > 0, \\ 1 \text{ sinon} \end{cases} \tag{4.1}$$

avec α le pourcentage de caractères bien détectés sur la colonne et γ la somme des pourcentages des autres éléments de la colonne.

En comparant les deux matrices de confusion, nous voyons qu'il n'y a pas beaucoup de changements sur les différentes classes entre avant et après l'ACP. Le '0' et le '1' restent les caractères les mieux

Classes	0	1	2	3	4	5	6	7	8	9	nbelem
0	**95**	0	0	0	0	1	1	0	1	1	413
1	0	**99**	0	0	0	0	0	0	0	0	470
2	1	0	**58**	6	7	9	4	6	6	4	429
3	0	0	6	**76**	0	3	4	0	8	1	401
4	0	0	7	0	**72**	2	2	3	1	13	370
5	1	1	9	6	3	**61**	4	6	4	3	361
6	1	0	2	2	1	1	**71**	4	1	17	364
7	0	1	2	0	1	2	7	**81**	1	4	401
8	2	0	1	4	2	1	2	0	**86**	3	395
9	1	1	1	2	3	2	13	2	2	**74**	396
Pertinence	0,94	0,97	0,67	0,79	0,81	0,74	0,66	0,79	0,78	0,62	

TABLE 4.6 – Matrice de confusion (en pourcentage de bonnes reconnaisssances), k-PPV et 39 descripteurs avec la base MNIST. Les colonnes correspondent aux valeurs reconnues, les lignes aux valeurs vraies.

Classes	0	1	2	3	4	5	6	7	8	9	Nombre d'éléments
0	**96**	0	0	0	0	0	2	0	1	1	395
1	0	**98**	0	0	0	0	0	0	0	0	425
2	1	0	**60**	7	8	7	3	4	8	3	410
3	0	0	6	**69**	0	4	5	2	11	2	416
4	0	0	6	0	**71**	2	2	3	2	14	382
5	1	0	10	6	2	**59**	3	8	5	6	374
6	1	0	1	1	1	0	**73**	3	1	20	364
7	0	0	1	0	3	1	6	**85**	0	3	409
8	2	0	1	3	3	0	2	0	**84**	4	415
9	1	1	2	0	1	2	10	1	2	**80**	410
Pertinence	0,94	0,99	0,69	0,8	0,8	0,79	0,69	0,8	0,74	0,6	

TABLE 4.7 – Matrice de confusion (en pourcentage de bonnes reconnaisssances), k-PPV après ACP conservant 20 axes avec la base MNIST. Les colonnes correspondent aux valeurs reconnues, les lignes aux valeurs vraies.

reconnus et les plus pertinents. Cela est dû à leur forme très spécifique. Par contre le '6' et le '9' sont assez souvent confondus du fait de l'invariance par rotation des descripteurs.

4.2.3 Base GrAnc

La table 4.8 donne les résultats de classification avec les 19 descripteurs de Zernike et les 10 moments de Fourier complexes, ensembles et après ACP conservant 10 et 20 axes.

Type de Descripteur	**Tous les descripteurs**	**Après ACP**	
Nombre de moments	29	10	20
KppvEucl			
Erreur	13,7%	13,2%	11,0%
Ecart type	2,0%	2,7%	1,3%

TABLE 4.8 – Combinaison de 10 moments de Fourier complexes et 19 moments de Zernike avec la base GrAnc

Nous notons une amélioration sensible des performances en combinant les descripteurs, le score *a priori* inférieur de l'ACP avec 20 axes est à relativiser car les écarts types sont relativement importants.

4.3 Commentaires

Les tests réalisés ici ont pour objectif de donner des ordres de grandeur de résultats de reconnaissance par rapport à nos bases de caractères. Ils nous permettent de nous rendre compte que les approches statistiques sont relativement efficaces et que les descripteurs de Zernike sont un excellent moyen de décrire ce genre de formes. Nous confirmons les performances médiocres des descripteurs basés sur les moments de Hu et de Flusser déjà constatées pour la reconnaissance de formes en général. Enfin la combinaison des moments de Zernike et des moments de Fourier complexes semble apporter un plus dans les scores de reconnaissance et l'utilisation de l'ACP permet en même temps de garder une dimension raisonnable.

Nous observons également que les scores de reconnaissance diminuent avec la complexité de la base. Pour la base GrCor qui est une base relativement propre avec peu de scripteurs les scores sont très bons par contre pour les deux autres bases les résultats sont plus mitigés. Ce résultat, certes attendu, permet de qualifier les bases utilisées. Ainsi, si nous nous fions aux résultats de reconnaissance avec les 29 descripteurs de Zernike et Fourier combinés et le k-PPV, il semble que la base de caractères la plus complexe soit la base MNIST, suivie de la base GrAnc puis de la base GrCor.

Maintenant que nous avons fixé des ordres de grandeur avec quelques méthodes statistiques et probabilistes classiques, nous allons nous intéresser à une autre grande famille de méthodes de reconnaissance, les approches dites structurelles.

Deuxième partie

Les approches structurelles

CHAPITRE 5

LA DESCRIPTION STRUCTURELLE D'UNE FORME

Sommaire

Partant de l'idée de HEBB ([Heb49]) que toute forme est appréhendée par la perception visuelle humaine, non dans sa globalité, mais par parties, et que l'organisation et les positions spatiales relatives de ces parties jouent un rôle important dans l'apprentissage et la reconnaissance, les descriptions structurelles se basent sur une décomposition des formes en éléments simples.

Appliquée à la reconnaissance de caractères, la notion d'élément simple renvoie à deux concepts différents. Un caractère peut se décomposer en graphèmes (en anglais « graphemes ») ou en segments primitifs (en anglais « strokes »). La littérature emploie bien souvent les deux termes avec confusion. Nous postulerons qu'un graphème est une composante connexe du caractère (le caractère entier si celui-ci n'est

composé que d'une composante connexe) et que les segments primitifs sont les éléments « géométriquement les plus simples possible » constituant un graphème. La recherche d'une entité « géométriquement simple » se comprend comme la volonté de ne pas faire entrer dans les structures utilisées des attributs géométriques trop complexes (comme les attributs statistiques par exemple). Puisque les descripteurs structurels reposent sur l'idée que les éléments simples sont interprétables par l'Homme, cette démarche est justifiée.

Ce chapitre présente les différentes étapes de construction des descripteurs structurels. Chaque étape est détaillée et illustrée par nos choix mis en œuvre dans le système de reconnaissance à base de graphes aléatoires que nous détaillerons ensuite dans le chapitre suivant.

Une forme géométrique simple, permettant une interprétation visuelle, est le segment. Or, pour décomposer un caractère en segments, ou en éléments de complexité comparable, il convient de schématiser la forme. C'est la squelettisation que nous aborderons dans le premier paragraphe de ce chapitre. La décomposition du squelette en segments primitifs sera l'objet du second paragraphe. Enfin nous terminerons par une présentation des structures utilisées en reconnaissance de forme.

5.1 La squelettisation

Un des problèmes fondamentaux en reconnaissance de formes est la représentation synthétique de celle-ci. Dans de nombreux cas le travail sur la forme brute est laborieux et inutile. Il est beaucoup plus avantageux en terme de temps et de qualité de travailler sur une forme épurée. La notion de squelette a été introduite à cet effet.

Dans le plan continu, le squelette d'une forme est un ensemble de lignes passant « en son milieu ». C'est la notion d'*axe médian* d'une forme continue introduite par BLUM en 1964 (dans une première version de l'article de 1967 [Blu67]). Il utilise pour cela l'analogie du feu de prairie. Elle offre une vision intuitive de la squelettisation : « *Soit une prairie couverte de manière homogène par de l'herbe sèche et* Ω *un ensemble de points de cette prairie. Si tous les points du contour de* Ω *sont enflammés simultanément, le feu se propage de manière homogène et s'étend à travers la prairie à une vitesse constante. Le squelette de* Ω *(noté MA(*Ω*)) est alors défini comme le lieu des points où les fronts enflammés se rencontrent.* ». L'avantage de cette définition est qu'elle peut être modélisée physiquement par une équation de diffusion du type :

$$\frac{\partial \mathcal{C}}{\partial t} = \beta \mathcal{N} \tag{5.1}$$

avec $\mathcal{C}$ la courbe paramétrée du contour, $\mathcal{N}$ la normale intérieure en chaque point et β la vitesse de diffusion. ROSENFELD ET PFLATZ ([RP66]) définissent, à la même période, l'axe médian avec la notion de boule maximale (définitions 6 et 7).

Définition 6 (Boule Maximale) *Une* boule B *incluse dans un objet* S *est dite* maximale *s'il n'existe pas d'autre boule* B' *incluse dans* S *et la contenant strictement.*

Définition 7 (Axe Médian, ROSENFELD ET PFLATZ) *L'*axe médian *de S est l'ensemble des centres des boules maximales de S (figure 5.1 page ci-contre).*

Définition 8 (Axe Médian Pondéré, ROSENFELD ET PFLATZ) *L'*axe médian pondéré *de S est l'ensemble des couples {centre,rayon} des boules maximales de S (figure 5.1 page suivante).*

Puis PAVLIDIS ([Pav82]) en donne une définition formelle équivalente (définition 9 page ci-contre).

FIGURE 5.1 – Squelette comme centre des boules maximales.

Définition 9 (Axe Médian, PAVLIDIS) *Soit R une forme plane, B son contour et P un point de R. Un* plus proche voisin *de P sur B est un point M de B tel qu'il n'existe aucun autre point sur B dont la distance à P est inférieure à la distance PM. Si P a plus d'un plus proche voisin, P est appelé* point squelette *de R. L'union de tous les points squelette est appelé* squelette *ou* axe médian *de R.*

Malgré ce formalisme, l'éventail des possibilités est d'autant plus vaste qu'il existe une multitude de définitions de ce qu'est une distance discrète. HILDICH propose de définir l'axe médian par ses propriétes ([Hil69]) :

Définition 10 (Axe Médian, propriétés d'après HILDICH)

1. *Epaisseur : le squelette est mince. Idéalement, il n'est constitué que de lignes ayant comme épaisseur un pixel.*
2. *Position : le squelette est « idéalement » positionné au centre des formes. Malheureusement le centre d'une forme discrète n'est pas unique.*
3. *Connectivité ou homotopie : l'axe médian possède exactement le même nombre de composantes connexes et pour chaque composante, le même nombre de trous que la forme initiale.*
4. *Stabilité : dans le cas d'un calcul du squelette par amincissement, tout ou partie du squelette ne doit pas pouvoir être érodé à nouveau sitôt qu'il ou elle est arrivé à stabilité. Autrement dit, sitôt qu'un pixel est considéré comme point squelette, on ne doit plus pouvoir remettre cette désignation en question. Cette propriété évite les érosions de squelette par les extrémités.*

Deux autres propriétés sont généralement ajoutées aux quatre précédentes ([Att95]) :

Définition 11 (Axe Médian, propriétés supplémentaires)

1. *Invariance par translation et rotation.*
2. *Réversibilité de l'axe médian pondéré (définition 8 page précédente), la forme doit pouvoir être reconstruite à partir du squelette et des rayons des boules maximales.*

Il est admis que ces conditions sont pour la plupart contradictoires ([BM99a]). Par exemple on ne peut pas avoir un squelette homotopique qui respecte la condition d'épaisseur stricte (1 pixel de largeur). Il faut donc faire des compromis et une comparaison de différents algorithmes est nécessaire pour choisir celui qui convient le mieux à une application donnée. Il existe globalement quatre approches algorithmiques pour le calcul de l'axe médian d'une forme binaire ([TK03]) : les méthodes d'amincissement topologique, les méthodes par cartes de distances, les méthodes par évolution de courbe et les approches géométriques basées sur le diagramme de Voronoï. En reconnaissance de caractères, les formes sont minces et le squelette est utilisé comme un résumé interprétable de la courbe. C'est pourquoi les méthodes rapides qui respectent la condition d'épaisseur stricte sont préférées aux méthodes strictement homotopiques et stables bien souvent plus lentes. De ce fait, les approches par amincissement sont privilégiées et nous restreindrons volontairement notre exposé à ces seules techniques.

Après avoir donné quelques définitions et explications générales sur les méthodes d'amincissement, nous exposerons quelques algorithmes classiques séquentiels et parallèles (principalement les algorithmes référencés par [LLS92]) et nous les comparerons sur quelques exemples de caractères.

5.1.1 Les méthodes d'amincissement

Les algorithmes d'amincissement examinent les pixels de l'image binaire et effacent itérativement ceux qui n'appartiennent pas au squelette final. Cet examen des pixels est fait de manière séquentielle ou parallèle. Dans une approche séquentielle, les pixels sont examinés pour être supprimés dans un ordre préétabli et fixe à chaque itération (ligne par ligne de gauche à droite par exemple). La suppression d'un point P à la n^{ieme} itération dépend des opérations faites jusqu'ici, c'est à dire à la $(n-1)^{ieme}$ itération mais aussi sur les pixels déjà traités à la n^{ieme}. Dans une approche parallèle, les pixels sont examinés de façon indépendante à chaque itération. La suppression de P à la n^{ieme} itération ne dépend que des opérations réalisées jusqu'à l'itération précédente.

5.1.1.1 Définitions et notations

Le choix des pixels à supprimer lors d'une itération dans un algorithme d'amincissement est délicat. Intuitivement, il semble naturel de supprimer les points de contour qui ne changent pas la connectivité. Il nous faut donc définir ce qu'est un point de contour et comment mesurer la connectivité d'un point.

X4	X3	X2
X5	P	X1
X6	X7	X8

FIGURE 5.2 – Pixels de N(P)

Par convention, nous noterons le voisinage $N(P)$ d'un point P de l'image comme explicité sur la figure 5.2. De plus, nous conviendrons, naturellement, que les objets d'interêt d'une image binaire sont représentés par les pixels de valeur 1 et que, par conséquent, les pixels de valeur 0 représentent le fond ou les trous. Si S désigne l'ensemble des points objets, $\bar{S}$, le complément de S, désigne l'ensemble des points de fond. Nous considérerons ensuite que P est une forme définie en 8-connexité et que son complément $\bar{P}$ est 4-connexe (afin de respecter le théorème de Jordan). Nous noterons, enfin, $b(P)$ le nombre de pixels noirs, donc à 1, dans $N(P)$.

Définition 12 (Point de contour) *Un point de contour est un point qui a au moins un pixel blanc dans son 4-voisinage. Si le pixel blanc est x_5 le point est un* point de contour gauche ou ouest, *s'il s'agit de x_1 le point est un* point de contour droit ou est, *pour x_3 c'est un* point de contour haut ou nord *et enfin pour x_7 il s'agit d'un* point de contour bas ou sud.

Définition 13 (Point de fin) *Un point de fin est un point tel que $b(P) = 1$.*

Les algorithmes d'amincissement vont donc chercher à mettre à 0 les points de contour qui ne sont pas des points de fin.

Définition 14 (8-(4-)composante) *Un sous-ensemble Q d'une image binaire I, est une* 8-(4-)composante *de I si et seulement si pour toute paire de points (P, M) de Q, il existe un 8-(4-)chemin allant de P à M.*

Définition 15 (8-(4-)connectivité) *La 8-(4-)connectivité d'un sous-ensemble S de l'image est le nombre de 8-(4-)composantes de son complément $\bar{S}$.*

Définition 16 (point 8-(4-)supprimable) *Un point P non nul est dit 8-(4-)supprimable si et seulement si sa mise à valeur nulle ne change pas la 8-(4-)connectivité de l'objet d'intérêt auquel il appartient. Un point 4-supprimable est 8-supprimable mais l'inverse n'est pas vrai (figure 5.3 page suivante).*

L'amincissement d'une forme consiste à supprimer les points de contour qui ne sont pas des points de fin et qui sont 8-(4-)supprimables (suivant le choix de la connectivité à conserver). Pour savoir si un point est 8-(4-)supprimable, il faut définir un critère calculable sur son voisinage. Ce critère est appelé, selon les auteurs :

– nombre d'intersections,
– nombre de connectivité,
– simplicité du pixel.

C'est RUTOVITZ qui commence par définir et utiliser le nombre d'intersections qui porte son nom ([Rut66]).

Définition 17 (Nombre d'intersections de RUTOVITZ)

$$X_R(P) = \sum_{i=1}^{8} |x_{i+1} - x_i| \tag{5.2}$$

avec $x_9 = x_1$.

$X_R(P)$ représente le nombre de transitions $0 \rightarrow 1$ et $1 \rightarrow 0$. Il est égal à deux fois le nombre de 4-composantes dans $N(P)$. Ainsi si $X_R(P) = 2$, il est certain que les pixels noirs de $N(P)$ sont 4-connexes (si $b(P) > 1$). Dans ce cas, et si P est un point de contour sans être un point de fin ($2 \leq b(p) \leq 6$), il est supprimable sans risque de modification pour la connectivité de la forme. En revanche, comme le critère est basé sur la 4-connectivité, les points 8-supprimables et non 4-supprimables ne seront pas supprimés. Il faut ajouter un post-processing spécifique pour ces points, ce qui est assez lourd.

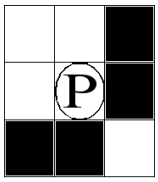


FIGURE 5.3 – Exemple de pixel 8-supprimable et non 4-supprimable

Afin de résoudre ce problème, HILDITCH a défini un nouveau nombre d'intersections basé, cette fois-ci, sur le nombre de 8-composantes dans $N(P)$, ([Hil69]).

Définition 18 (Nombre d'intersections de HILDITCH)

$$\begin{array}{l} X_H(P) = \sum_{i=1}^{4} b_i \\ b_i = \left\{ \begin{array}{l} 1\, si\, x_{2i-1} = 0\, et\, (x_{2i} = 1\, ou\, x_{2i+1} = 1) \\ 0\, sinon \end{array} \right. \end{array} \tag{5.3}$$

$X_H(P)$ est égal au nombre de 8-composantes de $N(P)$ excepté quand tous les 4-voisins de P sont noirs mais dans ce cas P n'est pas un point de contour. Ainsi si $X_H(P) = 1$, la suppression de P ne change pas la 8-connexité de l'objet. Enfin, et afin d'homogénéiser ces deux critères, YOKOI ET COLL. ont défini les nombres de Yokoi en 8 et 4-connexité ([YTF75]).

Définition 19 (Nombre de connectivité de YOKOI ET COLL.)

$$\begin{array}{l} C_8(P) = \sum_{i=1}^{4} (x_{2i-1} - x_{2i-1}\bar{x_{2i}}\bar{x_{2i+1}}) \\ C_4(P) = \sum_{i=1}^{4} (x_{2i-1} - x_{2i-1}x_{2i}x_{2i+1}) \end{array} \tag{5.4}$$

$C_8(P)$ est strictement équivalent à $X_H(P)$, en revanche $C_4(P)$ représente strictement le nombre de 4-composantes dans $N(P)$. Le critère $C_8(P)$ peut également être vu comme le nombre de fois où P sera traversé lors d'un suivi de contour, on parle de *simplicité*.

Définition 20 (Simplicité) *Un point P est dit* multiple *(ou de point de rupture) si $X_H(P) > 1$ et* simple *si $X_H(P) = 1$.*

Si P est un point multiple, sa suppression change la 8-connexité de l'objet, et seuls les points simples sont supprimables. Une autre notion souvent rencontrée est le point trou :

Définition 21 (Points trous) *Si $C_8(P) = 0$ et $b(P) > 1$, les quatre 4-voisins de P sont noirs et P est un* point trou.

D'une manière générale ([Tam78]), si on note CC le critère de connectivité utilisé qui peut être $C_8(P)$, $C_4(P)$, $X_H(P)$ ou encore $X_R(P)/2$, le point P considéré est :
- un point intérieur ou un point trou si $CC = 0$,
- un point simple si $CC = 1$,
- un point multiple de connexion si $CC = 2$,
- un point multiple de bifurcation (*branching point*) si $CC = 3$,
- un point multiple de croisement (*crossing point*) si $CC = 4$.

En résumé, les points qui peuvent être marqués comme supprimables par un algorithme d'amincissement sont les points noirs simples. Tous les autres ne sont pas supprimables sous peine de perdre la connectivité de la forme de départ. La difficulté réside ensuite dans le balayage de l'image et la sélection des points supprimables à chaque itération, sous les conditions d'épaisseur, de position et de stabilité de la définition 10 page 63. En effet, la suppression simultanée des points simples peut provoquer des instabilités et fausser l'homotopie. Il convient donc d'ajouter des conditions sur les voisinages des points simples pour les supprimer. Sur ce point, les avis divergent et la multitude des articles et algorithmes prouve qu'il n'y a pas de recette et que le squelette est finalement dépendant de l'application. Un article très intéressant sur ce sujet est celui de LEE ET COLL. ([PSBB93]) dans lequel trois méthodes différentes d'évaluation de la qualité d'un jeu de squelettes sont explicitées. L'une d'entre elles est basée sur l'appréciation de 33 personnes et montre, entre autres, que la condition d'épaisseur stricte est importante pour un lecteur de squelettes de caractères.

Dans toutes les approches qui suivent, les pixels retenus pour être supprimés sont considérés comme simples, suivant un des quatre critères énoncés ci-avant. Toutefois le critère pourra être reprécisé dans les conditions de marquage par fidélité aux énoncés des articles.

5.1.2 Algorithmes séquentiels d'amincissement

Les approches séquentielles ont pour avantage de traiter plus de pixels par itération. Mais ceci se fait au détriment de la stabilité et de l'homotopie, puisque la suppression d'un point change le voisinage du suivant. C'est pourquoi nous avons opté pour une approche parallèle. Cependant nous donnons ici deux algorithmes historiques qui nous ont servi de comparatif.

5.1.2.1 Algorithme de HILDITCH

En définissant son nombre d'intersections HILDITCH a proposé une appoche séquentielle pour réaliser l'amincissement. L'image est balayée de gauche à droite et de haut en bas puis les pixels qui répondent simultanément aux quatre critères suivants sont marqués :

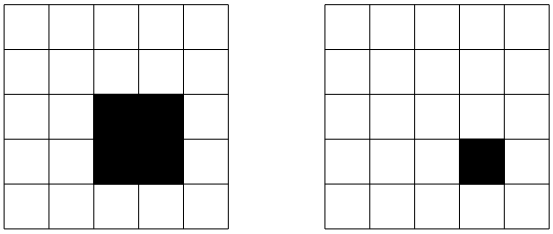
H1 : Au moins un voisin noir de P n'est pas marqué,

H2 : $X_H(P) = 1$,

H3 : si x_3 est marqué, $x_3 = 0$ ne change pas $X_H(P)$,

H4 : si x_5 est marqué, $x_5 = 0$ ne change pas $X_H(P)$,

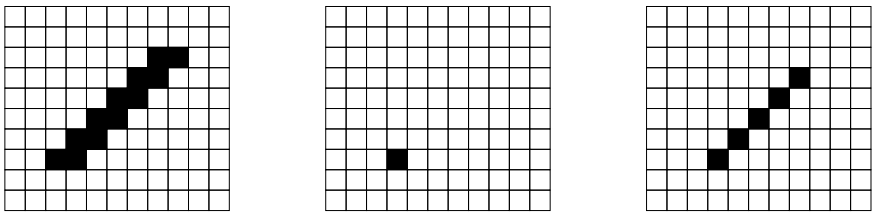
La condition H1 permet d'éviter l'érosion des petits ensembles circulaires, elle en assure la stabilité. Ainsi sur la figure 5.4, si la condition H1 n'existait pas, le carré serait entièrement supprimé à la première itération.



(a) (b)

FIGURE 5.4 – Exemple de l'interêt de H1, (a) avant amincissement, (b) après amincissement de HILDITCH

Les conditions H3 et H4, quant à elles, préservent la stabilité des lignes doubles comme nous pouvons le voir sur la figure 5.5.



(a) (b) (c)

FIGURE 5.5 – Exemple de l'interêt de H3 et H4, (a) avant amincissement, (b) après amincissement de HILDITCH sans les conditions H3 et H4, (c) après amincissement de HILDITCH avec les conditions H3 et H4

Le problème principal de cette approche est le temps de calcul. Pour rendre l'algorithme plus rapide, HILDITCH propose une généralisation avec des fenêtres de dimension $k \times k$ avec $k \geq 3$. Il va chercher à supprimer tous les pixels du centre de la fenêtre $(k-2) \times (k-2)$. Si les pixels du bord de la fenêtre sont tels que $X_H(P) = 1$ et s'ils contiennent plus de $(k-2)$ pixels blancs 4-connectés et plus de $(k-2)$ pixels noirs, alors les pixels du centre de la fenêtre sont supprimables. Sinon la taille de la fenêtre est réduite à $(k-1) \times (k-1)$, et ainsi de suite. Avec des grandes valeurs de k, le squelette peut s'obtenir en une itération. Cependant plus k est grand et plus le squelette obtenu est grossier et bruité.

5.1.2.2 Algorithme de YOKOI ET COLL.

Dans l'article ([YTF73]), YOKOI ET COLL. présentent la règle de marquage des pixels suivante :

$$L(P) = \begin{cases} 2\, si\, x_3 = 0\, ou\, \bar{x_3} + \bar{x_5} + x_7 = 0 \\ 3\, si\, \bar{x_3} + x_5 = 0\, ou\, \bar{x_3} + \bar{x_5} + \bar{x_7} + x_1 = 0 \end{cases} \tag{5.5}$$

A chaque itération, un premier balayage de l'image permet de marquer les pixels dont le critère est égal à 2 ou 3. Les pixels marqués 2 sont supprimés en balayant l'image de haut en bas et de gauche à droite. Enfin les pixels marqués 3 sont supprimés de la même façon en balayant, cette fois-ci, l'image de bas en haut et de droite à gauche. Pour les algorithmes séquentiels, le sens de balayage est primordial

puisque la suppression d'un point va tenir compte de la suppression des points précédents.

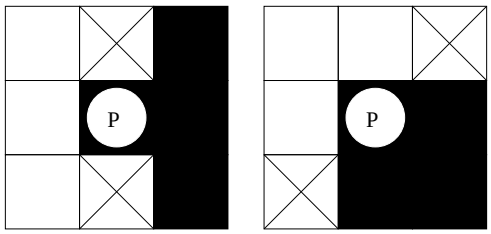
5.1.3 Algorithmes parallèles d'amincissement

Dans ce type d'algorithmes, les points sont examinés selon le résultat de la dernière itération seulement. Cela permet de paralléliser les examens à chaque itération. Ces algorithmes sont par contre plus lents que les algorithmes séquentiels sur des machines séquentielles puisqu'ils suppriment moins de pixels à chaque itération.

Les algorithmes parallèles, à l'inverse des algorithmes séquentiels, préservent mieux les propriétés de stabilité et d'homotopie. Mais ils nécessitent bien souvent l'utilisation de sous-itérations à la fin desquelles l'image est mise à jour pour la sous-itération suivante. Bien souvent une sous-itération pour chaque type de point de contour (Ouest, Sud, Est, Nord) est utilisée, ou deux sous-itérations, une pour les points Est et Nord, et une autre pour les points Ouest et Sud.

5.1.3.1 L'algorithme morphologique de GONZALES ET WOOD

La première approche intuitive d'amincissement parallèle est celle de GONZALES ET WOOD ([GW92]) qui proposent un algorithme en 1 itération. Elle consiste à marquer comme supprimables les pixels dont le voisinage correspond à une des configurations $B_i, i = 1, \ldots, 8$ de la figure 5.6 représentant en fait les points de contours dont $3 \leq b(P) \leq 5$. L'implémentation proposée utilise l'opération morphologique « Tout ou Rien » (voir annexe C page 169) qui n'est pas optimale puisqu'elle impose un balayage de toute l'image pour chaque B_i.



(B_1) (B_2)

FIGURE 5.6 – Les masques de GONZALES ET WOOD ($B_3, \ldots, B_8$ sont obtenus par toutes les rotations de 90°), les pixels en croix peuvent prendre la valeur 0 ou 1.

Cet algorithme présente beaucoup d'inconvénients. Il est extrêmement sensible aux bruits et relativement lent dans l'implémentation proposée. De plus, ne considérant que les points simples tels que $3 \leq b(P) \leq 5$, il a tendance à ajouter des aspérités.

5.1.3.2 L'algorithme de RUTOVITZ

RUTOVITZ ([Rut66], [SR71]), balaye l'image et supprime le pixel si et seulement si toutes les conditions suivantes sont vraies :

R1 : $2 \leq b(P) \leq 6$

R2 : $X_R(P) = 2$

R3 : $x_1 x_3 x_5 = 0 \, ou \, X_R(x_3) \neq 2$

R4 : $x_1 x_3 x_7 = 0 \, ou \, X_R(x_1) \neq 2$

La condition R1 assure que P est un point de contour et que sa suppression ne provoquera pas de trou. Cet algorithme est simple car il ne nécessite qu'une sous-itération. En revanche, il ne prend pas en compte la simplicité des points de voisinage pour les points supprimables, il entraîne donc des collisions non gérées. Ceci implique que la stabilité du squelette n'est pas assurée (souvent trop érodé) et conduit à un squelette décalé par rapport à l'axe médian. Il ne réduit pas non plus les lignes doubles 8-connexes.

5.1.3.3 L'algorithme de DEUTSCH

DEUTSCH ([Deu72]) a ajouté une condition à celles de RUTOVITZ pour résoudre le problème des lignes doubles 8-connexes. Puis il a ajouté une sous-itération en effectuant une rotation de 180°sur les conditions de suppression pour tenter d'aligner le squelette avec l'axe médian. Cela donne les conditions suivantes :

- Première sous-itération :

  D1 : $X_R(P) = 0$, 2 ou 4

  D2 : $b(P) \neq 1$

  D3 : $x_1 x_3 x_5 = 0$

  D4 : $x_1 x_3 x_7 = 0$

  D5 : si $X_R(P) = 4$ alors une des conditions (a) ou (b) suivantes doit être vraie :

  (a) : $x_1 x_7 = 1$, $x_2 + x_6 \neq 0$, et $x_3 + x_4 + x_5 + x_8 = 0$

  (b) : $x_1 x_3 = 1$, $x_4 + x_8 \neq 0$, et $x_2 + x_5 + x_6 + x_7 = 0$

- Deuxième sous-itération :

  D1b : $X_R(P) = 0$, 2 ou 4

  D2b : $b(P) \neq 1$

  D3b : rotation de 180° par rapport à D3 c.-à-d. $x_1 x_5 x_7 = 0$

  D4b : rotation de 180° par rapport à D4 c.-à-d. $x_3 x_5 x_7 = 0$

  D5b : rotation de 180° par rapport à D5 c.-à-d. si $X_R(P) = 4$ alors une des conditions (a') ou (b') suivantes doit être vraie :

  (a') : $x_3 x_5 = 1$, $x_2 + x_6 \neq 0$, et $x_1 + x_4 + x_7 + x_8 = 0$

  (b') : $x_5 x_7 = 1$, $x_4 + x_8 \neq 0$, et $x_1 + x_2 + x_3 + x_6 = 0$

Là encore, les collisions ne sont pas gérées mais leurs conséquences sont minimisées par la double itération ; en revanche, les carrés sont souvent complètement érodés.

5.1.3.4 L'algorithme de ZHANG ET SUEN

Cet algorithme ([ZS84]) est le plus souvent cité et utilisé. Il consiste en deux sous-itérations pour repérer les pixels à effacer selon les critères suivants :

- Première sous-itération :

  Z1 : $2 \leq b(P) \leq 6$

  Z2 : $X_R(P) = 2$

  Z3 : $x_1 x_3 x_7 = 0$

  Z4 : $x_1 x_5 x_7 = 0$

- Deuxième sous-itération :

  Z1b : idem Z1

  Z2b : idem Z2

Z3b : rotation de 180° par rapport à Z3 c.-à-d. $x_3x_5x_7 = 0$

Z4b : rotation de 180° par rapport à Z4 c.-à-d. $x_1x_3x_5 = 0$

Cet algorithme est relativement efficace et robuste au bruit de contour. Néanmoins certaines parties du squelette peuvent être décalées par rapport au centre.

5.1.3.5 L'algorithme de ZHANG ET WANG

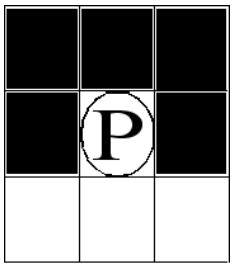


FIGURE 5.7 – Exemple de collision : si on supprime x_3, P n'est plus simple.

ZHANG ET WANG ([ZW94]) présentent un algorithme parallèle en une seule itération (voir algorithmes 1 page suivante et 2 page 72) qui produit des squelettes parfaitement 8-connexes et qui gère les collisions. Pour se faire, ils font l'inventaire de toutes les collisions possibles (exemple figure 5.7) lors de la suppression d'un point simple (caractérisé par $X_H(P) = 1$ et $b(P) > 1$) suivant sa nature (Ouest, Est, Nord, Sud, Sud-Ouest, Nord-Ouest, Sud-Est, Nord-Est). Chaque point P se voit attribuer une étiquette de 0 à 8 suivant la configuration de ses voisins avec la fonction $WN\ :\ P \rightarrow \{0, \ldots, 8\}$ telle que :

$WN(P) = 0$ - P est un point multiple

$WN(P) = 1$ - P est un point simple

$WN(P) = 2$ - P est un point simple et x_3 est un point multiple

$WN(P) = 3$ - P est un point simple et x_5 est un point multiple

$WN(P) = 4$ - P est un point simple et x_3 et x_5 sont des points de multiple

$WN(P) = 5$ - P est un point simple et x_7 est un point multiple

$WN(P) = 6$ - P est un point simple et x_3 et x_7 sont des points multiple

$WN(P) = 7$ - P est un point simple et x_3 et x_5 et x_7 sont des points multiple

$WN(P) = 8$ - P est un point simple et x_2 est un point multiple

Cet algorithme donne de très bons résultats et reste relativement rapide.

5.1.3.6 L'algorithme MB2

BERNARD ET MANZARENA ([BM99a]) proposent également un algorithme parallèle en une itération qui calcule le squelette non mince mais totalement homotopique en gérant toutes les collisions. Nous nous en servirons comme base de comparaison. Ils partent du principe fondamental suivant lequel un algorithme de squelettisation préserve parfaitement la k-connexité si et seulement si les points supprimés à chaque itération forment un ensemble k-simple (c.-à-d. que toute suppression de toute ou partie des points de l'ensemble, quel que soit l'ordre de suppression, ne change pas la connexité de la forme). La difficulté réside évidemment dans le fait qu'un ensemble de points simples n'est pas obligatoirement simple. Mais RONSE ([Ron88]) a proposé une condition suffisante pour caractériser un ensemble k-simple dont BERNARD ET MANZARENA dérivent le théorème 3 page 72.

Algorithme 1: Fonction SimplePoint

Données : Un objet du type pixel P
Résultat : Un booléen qui prend la valeur VRAI si le pixel est simple, FAUX sinon
début
  $point \leftarrow$WN(P);
  $north \leftarrow$WN(x_3);
  $east \leftarrow$WN(x_1);
  $west \leftarrow$WN(x_5);
  $northeast \leftarrow$WN(x_8);
  $SimplePoint \leftarrow FAUX$;
  suivant $point$ **faire**
    cas où = *1*
      $SimplePoint \leftarrow VRAI$
    cas où = *2*
      si $north$ = *0* **alors**
        $SimplePoint \leftarrow VRAI$
      fin
    cas où = *3*
      si $west$ = *0* **alors**
        $SimplePoint \leftarrow VRAI$
      fin
    cas où = *4*
      si *(*$north$ = *0) et (*$west$ = *0)* **alors**
        $SimplePoint \leftarrow VRAI$
      fin
    cas où = *5*
      si $east$ = *0* **alors**
        $SimplePoint \leftarrow VRAI$
      fin
    cas où = *6*
      si *(*$north$ = *0) et (*$east$ = *0)* **alors**
        $SimplePoint \leftarrow VRAI$
      fin
    cas où = *7*
      si *(*$north$ = *0) et (*$west$ = *0) et (*$east$ = *0)* **alors**
        $SimplePoint \leftarrow VRAI$
      fin
    cas où = *8*
      si $northeast$ = *0* **alors**
        $SimplePoint \leftarrow VRAI$
      fin
  fin
fin

Algorithme 2: Programme principal pour le calcul du PPTA

Données : L'image binaire sous forme de tableau de pixels $Q(i,j)$ où $1 \leq i \leq nblines$ et $1 \leq j \leq nbcol$

Résultat : L'image binaire squelettisée

début

  $NothingChanged \leftarrow VRAI$;

  répéter

    pour $i \leftarrow 1$ ***à*** $nbline$ **faire**

      pour $j \leftarrow 1$ ***à*** $nbcol$ **faire**

        si $Q[i,j] > 0$ *et* `SimplePoint(`$Q[i,j]$`)` **alors**

          $Q[i,j] = 0$;

          $NothingChanged \leftarrow FAUX$;

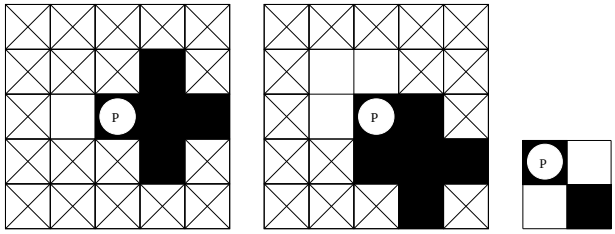
        fin

      fin

    fin

  jusqu'à $NothingChanged$;

fin



(α_1) (α_2) (β)

FIGURE 5.8 – Les masques de MB2 (avec les rotations de 90°), les pixels en croix peuvent prendre la valeur 0 ou 1.

Théorème 3 *Pour respecter la 8-(4-)connexité, il est suffisant que les algorithmes parallèles d'amincissement respectent simultanément les trois conditions suivantes :*

1. *tous les pixels supprimés sont simples ;*
2. *une 8-composante connexe contenue dans un carré 2×2 ne peut pas être complètement supprimée ;*
3. *toute paire de pixels 4-adjacents simultanément supprimables doit être supprimée séquentiellement.*

Ils démontrent ensuite que cette condition suffisante est équivalente à la suppression simultanée des pixels dont le voisinage est α_1 ou α_2 (figure 5.8, et leurs trois versions après rotation de 90°) mais qui n'est pas β (et ses trois versions après rotation de 90°).

Nous avons choisi par commodité une implémentation morphologique à base d'opérations « Tout ou Rien » (voir annexe C page 169) qui impose un parcours de l'image pour chaque masque. Il existe une version rapide de cet algorithme proposé par les auteurs.

5.1.4 Comparaisons et choix

Les résultats des méthodes de squeletisation sur différents caractères plus ou moins bruités sont présentés sur les figures 5.10 page 74, 5.11 page 75, 5.12 page 75, 5.13 page 76 et 5.14 page 76. Les images

Nom	Algorithme
HIL	HILDITCH, paragraphe 5.1.2.1 page 66
YOC4	YOKOI ET COLL., paragraphe 5.1.2.2 page 67, avec le critère C_4
YOC8	YOKOI ET COLL., paragraphe 5.1.2.2 page 67, avec le critère C_8
GON	GONZALES ET WOOD, paragraphe 5.1.3.1 page 68
RUT	RUTOVITZ, paragraphe 5.1.3.2 page 68
DEU	DEUTSCH, paragraphe 5.1.3.3 page 69
ZSU	ZHANG ET SUEN, paragraphe 5.1.3.4 page 69
ZWA	ZHANG ET WANG, paragraphe 5.1.3.5 page 70
MB2	MB2, paragraphe 5.1.3.6 page 70

TABLE 5.1 – Algorithmes

Images	'K'		'kutub'		'Kappa'		'Beta'		'Alpha'	
	It	t	It	t	It	t	It	t	It	t
HIL	6	4	28	2197	8	13	10	40	10	43
YOC4	15	8	81	4852	24	28	33	99	27	88
YOC8	15	8	84	5029	24	28	27	81	30	99
GON	80	24	256	7642	128	78	128	185	160	252
RUT	12	5	62	2077	18	15	22	41	26	53
DEU	12	4	48	1551	24	16	24	39	32	56
ZSU	16	6	80	2735	24	17	28	49	32	60
ZWA	10	12	34	333	16	11	14	20	20	24
MB2	125	57	400	16282	175	153	175	358	225	499

TABLE 5.2 – Nombre d'itérations (It) et temps de calcul (t en millisecondes)

sont traitées après fermeture morphologique par un masque 3×3 plein ([SM93]). La fermeture est utilisée comme prétraitement pour fermer les trous, existant à l'intérieur des caractères bruités, et pour relier les petites aspérités (figure 5.9 page suivante). Les noms des algorithmes sont explicités dans la table 5.1. Enfin la table 5.2 donne le nombre d'itérations et le temps de calcul pour chaque algorithme et pour chaque image. Rappelons que les performances de l'algorithme MB2 ne sont données qu'à titre d'information puisque l'implémentation morphologique utilisée n'est pas du tout optimale.

En terme de performances en temps de calcul, les deux algorithmes les plus rapides sont ceux de DEUTSCH et de ZHANG ET WANG. L'avantage est du côté de celui de ZHANG ET WANG qui est 5 fois plus rapide sur le traitement du mot arabe 'kutub'.

Le 'K' typographié de la figure 5.10 page suivante est un caractère de petite taille qui met en évidence les érosions trop importantes sur les extrémités. D'une manière générale, et par comparaison avec l'algorithme MB2, seul l'algorithme de GONZALES ET WOOD conserve bien les extrémités. Celui de ZHANG ET WANG est un peu trop érosif sur l'extrémité en haut à gauche, néanmoins c'est l'algorithme le plus fin et le mieux centré.

Le mot arabe typographié de la figure 5.11 page 75 met en évidence les algorithmes conduisant à des squelettes mal centrés et trop érodés. Là encore seuls ceux de DEUTSCH et de ZHANG ET WANG s'en

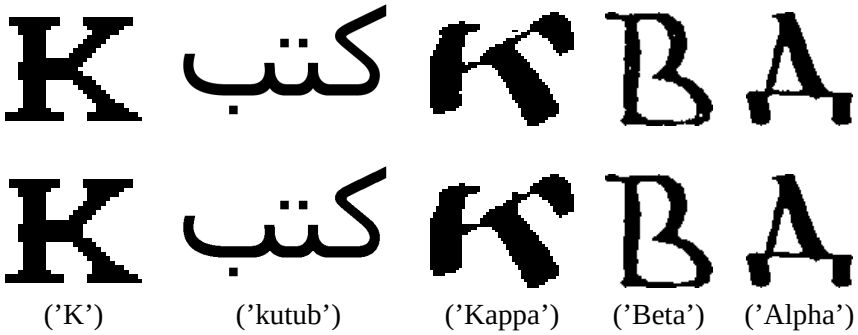


FIGURE 5.9 – Les images de test, caractères bruts sur la première ligne et après fermeture sur la deuxième ligne. Le 'K' et le 'kutub' sont typographiés, les autres sont manuscrits.

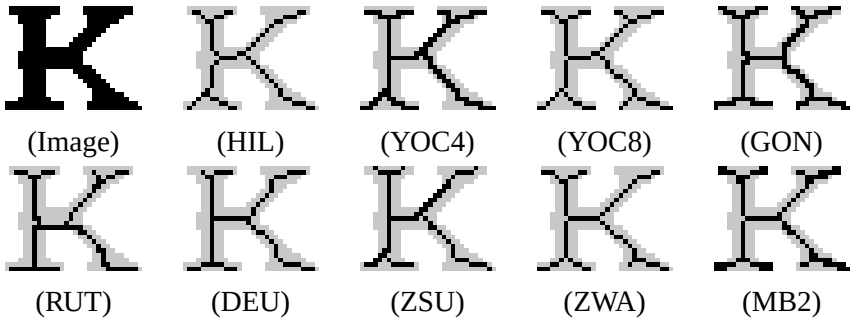


FIGURE 5.10 – Squelettes obtenus à partir du caractère 'K' typographié.

sortent bien avec un avantage pour ce dernier, il n'érode pas complètement les carrés.

Enfin sur les trois caractères grecs manuscrits, les algorithmes de DEUTSCH et de ZHANG ET WANG donnent des résultats plus robustes, plus précis et finalement pas trop érodés. Entre les deux, notre choix s'est porté sur l'algorithme de ZHANG ET WANG qui présente en plus l'avantage de donner un squelette parfaitement 8-connexe. Nous considérerons donc dans la suite de ce mémoire, et sauf précision contraire, que le terme de squelette, employé pour une forme, renvoie à cette méthode.

5.2 L'extraction des segments primitifs

Une fois le squelette d'un caractère obtenu, il nous faut en extraire les segments primitifs dont nous avons parlé au début de ce chapitre. Nous avions alors postulé qu'un segment primitif était un élément « géométriquement le plus simple possible » constituant un graphème. Le graphème étant maintenant schématisé par un squelette, une définition formelle du segment primitif peut être la suivante :

Définition 22 (Formalisme de la notion de segment primitif) *Nous appelons segment primitif tout ensemble de points strictement 8-connexes ayant au plus deux 8-voisins et formant une courbe de concavité stricte*[1] *dans le plan image.*

L'objectif d'une telle définition est bien de permettre une description géométrique des segments primitifs avec des attributs simples comme la courbure ou la longueur. Or ces grandeurs sont ambigües dès lors que la condition de concavité n'est pas respectée. A partir de cette définition 22, nous pouvons

[1] $\forall \{A, B\} \in \mathcal{C}, \,]AB[$ n'intersecte pas la courbe $\mathcal{C}$.

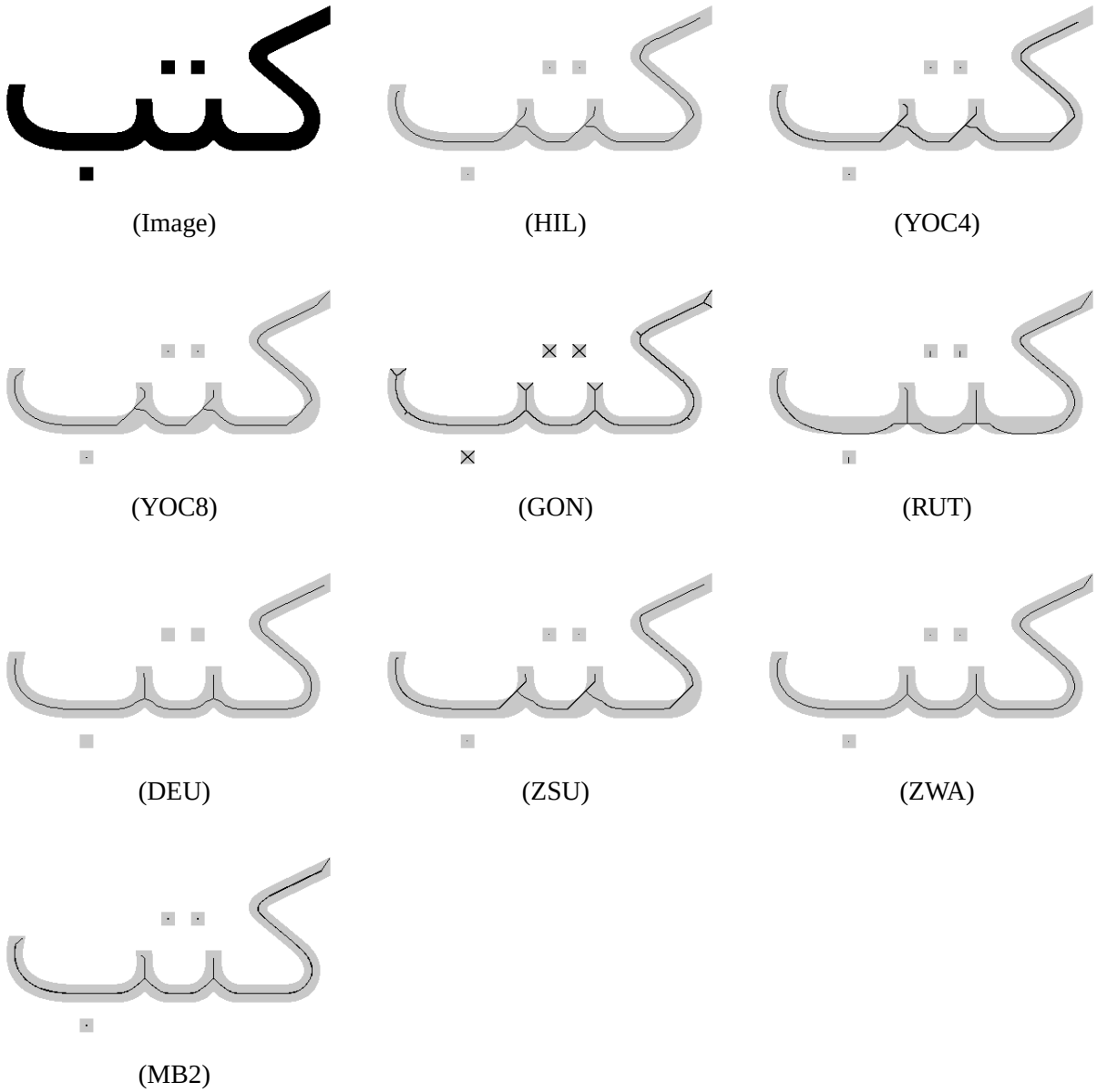


FIGURE 5.11 – Squelettes obtenus à partir du mot arabe ’kutub’, typographié.

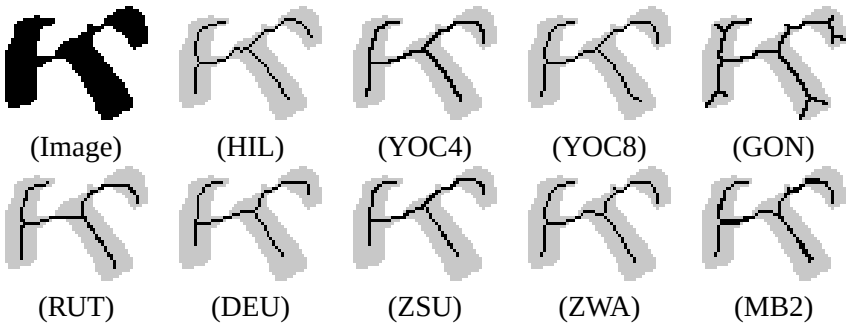


FIGURE 5.12 – Squelettes obtenus à partir du caractère ’Kappa’ manuscrit.

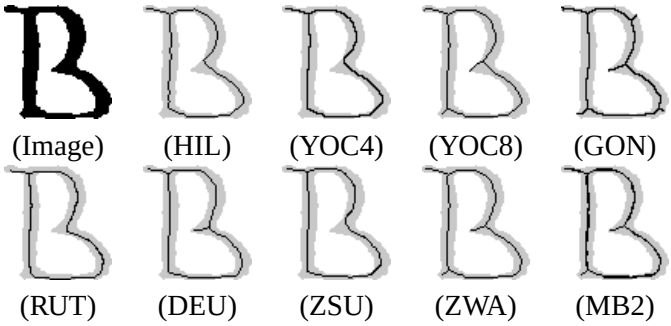


FIGURE 5.13 – Squelettes obtenus à partir du caractère 'Beta' manuscrit.

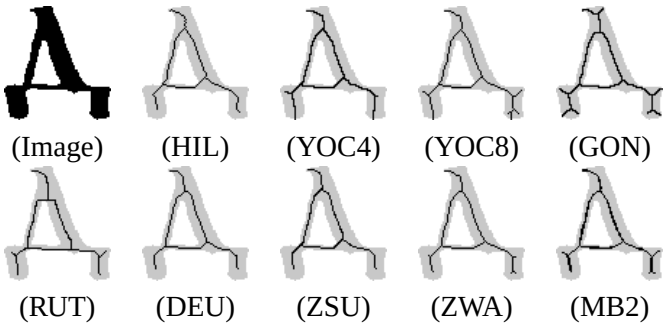


FIGURE 5.14 – Squelettes obtenus à partir du caractère 'Alpha' manuscrit.

maintenant considérer le squelette comme étant un ensemble de segments primitifs connectés entre eux par des points d'intersection (dont le nombre de 8-voisins est supérieur à 2) ou des points d'inflexion. La décomposition d'un squelette en segments primitifs se résume donc à l'extraction des points de fin (ayant un seul 8-voisin), des points d'intersection et des points d'inflexion qui vont composer les extrémités des segments primitifs.

5.2.1 Détection des points de fin et d'intersection

Cette étape ne pose pas de problème majeur puisqu'elle réside en une étude du voisinage de chacun des points. Les points de fin ne possèdent qu'un seul 8-voisin et les points d'intersection au moins trois. Une méthode de détection possible est de balayer l'image et de tester (par une 'et' logique) si les voisinages de chacun des points correspondent à un masque booléen caractéristique d'un point d'intersection ou de fin. Ainsi les points d'intersection correspondent aux masques suivants :

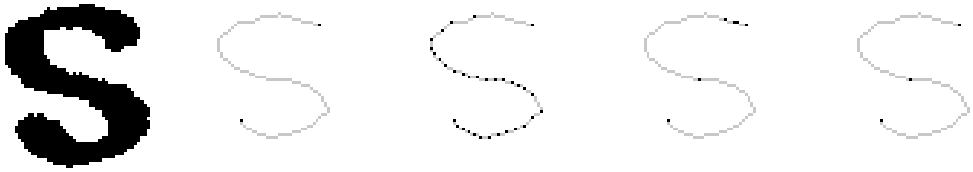
$$
\begin{aligned}
I^1 &= \begin{bmatrix} 1 & 0 & 1 \\ \times & 1 & 0 \\ 0 & \times & 1 \end{bmatrix} \text{ et rotations de 45°.} \\
I^2 &= \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ \times & 1 & \times \end{bmatrix} \text{ et rotations de 45°.} \\
I^3 &= \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \text{ et rotations de 45°.}
\end{aligned}
\tag{5.6}
$$

où le symbole $\times$ représente un 1 ou un 0, et les points de fin s'obtiennent, quant à eux, avec les masques suivants :

$$F^1 = \begin{bmatrix} 1 & 0 & 0 \\ \times & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \text{ et rotations de 45°.}$$
$$F^2 = \begin{bmatrix} 1 & \times & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \text{ et rotations de 45°.} \tag{5.7}$$

5.2.2 Détection des points d'inflexion

Après la détection des points de fin et d'intersection, le squelette peut être divisé en un ensemble de primitives dont la concavité n'est pas obligatoirement stricte. Les points d'inflexion se calculent alors primitive par primitive. Par définition, un point d'inflexion est le lieu où la courbure de la courbe s'annule (avec une gestion particulière des portions de segment). Un moyen simple consiste à utiliser une interpolation puis à trouver les lieux où $D = (x'' \times y' - x' \times y'') = 0$. L'algorithme de de Casteljau (dont une revue historique intéressante est donnée dans [BM99b]) est tout à fait approprié pusqu'il lisse les ruptures. Cependant l'utilisation de tous les points de la primitive comme points de controle implique une approximation trop bruitée et conduit à des détections parasites (figure 5.15 (d)). L'utilisation d'une polygonalisation de Ramer ([Ram72]) induit un lissage de la courbe. Grâce à cela, le nombre des points de contrôle se trouve limité, améliorant ainsi les résultats. Sur la figure 5.15, le squelette de la forme présenté en (a) est donné en (b) avec ses deux points de fin. L'image (c) donne les points de polygonalisation de Ramer qui vont servir ensuite de points de contrôle pour l'interpolation de de Casteljau. A partir de la courbe paramétrée issue de l'interpolation, l'image (e) présente le point d'inflexion obtenu (ainsi que les deux points de fin). L'image (d) donne, quant à elle, les points d'inflexions obtenus après une interpolation de de Casteljau utilisant tous les points de la courbe comme points de contrôle. Nous constatons que la polygonalisation permet de ne plus détecter les points parasites en haut du squelette.



(a) (b) (c) (d) (e)

FIGURE 5.15 – (a) image originale, (b) squelette et points de fin, (c) polygonalisation de Ramer, (d) points d'inflexion sans polygonalisation, (e) point d'inflexion avec polygonalisation.

5.3 Les structures

Les descriptions structurelles reposent sur une décomposition des formes en données symboliques. Elles tendent à modéliser les éléments « signifiants » ainsi que leurs relations. Autrement dit, elles cherchent à interpréter la forme en fonction d'un ensemble de primitives prédéterminées et liées par des relations d'adjacence, de topologie, d'inclusion...

Deux types de descriptions sont couramment utilisées en RdF. D'un côté, les descriptions unidimensionnelles, ou chaînes, permettent une reconnaissance syntaxique, stochastique ou graphique. De l'autre côté, les descriptions topologiques, graphes ou cartes, sont plus complètes mais restreignent le champ des techniques de reconnaissance possibles du fait de leur complexité.

5.3.1 Les chaînes

La chaîne est, par définition, une suite de symboles. Dans le cas de la RdF, les symboles sont des primitives et leur ordre dans la chaîne est souvent lié à leur voisinage dans l'image. Les structures de chaînes se prêtent relativement bien à la description des contours. Les primitives, qui sont alors basiquement les pixels et leurs relations, sont codées grâce au code de FREEMAN ([Fre61], figure 5.16 (a)). Chaque pixel est représenté dans la chaîne par l'entier qui désigne sa position par rapport au pixel précédent. La chaîne commence conventionnellement par les coordonnées du point de départ. Dans l'exemple de la figure 5.16 (b), le code de la chaîne de FREEMAN serait :
(3,3)70701013345321013432170113344545666566666766
Comme il s'agit d'une chaîne cyclique, les coordonnées du point de départ *(3,3)* peuvent être omises.

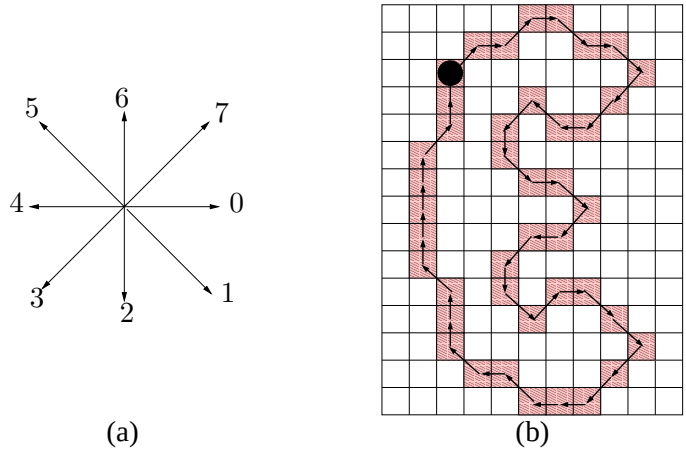


FIGURE 5.16 – Codage de Freeman.

Un tel codage de chaîne est dit *absolu* puisque les directions codées sont définies par rapport au cadre de l'image. Il n'est pas invariant par rotation. Pour obtenir une telle invariance, il faut utiliser un codage *relatif* qui utilise des directions définies par rapport au pixel voisin. Ce sont les différences de directions des pixels deux à deux qui seront codées. Pour obtenir un codage de FREEMAN relatif, il suffit de soustraire les codes absolus deux à deux. Ainsi, dans l'exemple de la figure 5.16 (b), le code relatif de la chaîne de FREEMAN serait :
1711712011677712177761102010171107100017 01.

D'autres types de chaînes, plus ou moins inspirées de celle de FREEMAN, sont également utilisées en reconnaissance de formes. Citons, entre autres, les *chaînes de sommets* développées par BRIBIESCA ([Bri99]) qui considère chaque sommet des pixels de contour comme un élément de la chaîne associé à un entier représentant le nombre de pixels du contour lui étant adjacents (figure 5.17 page ci-contre).

5.3.2 Les structures topologiques

La topologie est l'étude relative des propriétés des objets dans l'espace. Les structures topologiques s'intéressent à la description des relations des objets entre eux (inclusion, adjacence, relations d'incidence...). Il existe deux types de représentation qui eux-mêmes se dérivent en plusieurs variantes : les cartes combinatoires et les graphes.

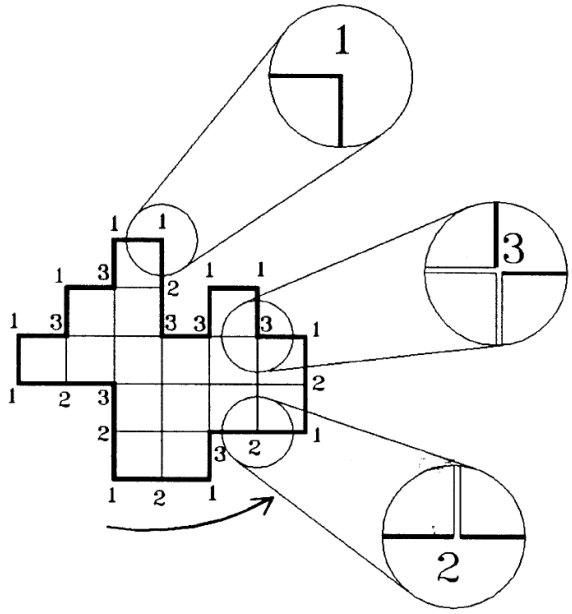


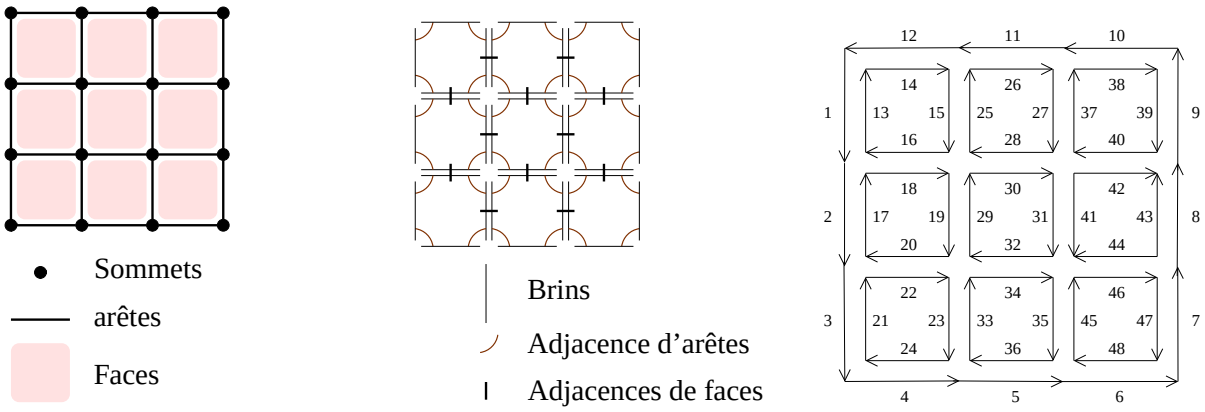
FIGURE 5.17 – Principe de codage des chaînes de sommets de BRIBIESCA([Bri99])

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	...
β_1	2	3	4	5	6	7	8	9	10	11	12	1	14	15	16	13	18	19	...
β_2	13	17	21	24	36	48	47	43	39	38	26	14	1	12	25	18	2	16	...

TABLE 5.3 – β_1 et β_2 de la figure 5.18

5.3.2.1 Les cartes combinatoires

Les cartes combinatoires ([DBF04]) sont définies pour structurer la topologie des espaces de dimension deux ou plus.



(a) (b) (c)

FIGURE 5.18 – Construction d'une carte combinatoire de dimension 2 (a) subdivision de l'espace topologique, (b) carte combinatoire schématique associée, (c) carte combinatoire formelle.

Formalisme et définitions Les *cartes combinatoires* sont utilisées pour la description complète d'un espace topologique de dimension quelconque, deux dans le cadre qui nous intéresse. La subdivision d'un espace topologique de dimension deux (voir figure 5.18 page précédente (a)) est une partition de l'espace en trois sous-ensembles dont les éléments sont des *cellules* de dimension 0, 1 et 2 (appelés respectivement *sommets*, *arêtes* et *faces* et notés i-*cellule* où i est la dimension de la cellule considérée). Le *bord* d'une i-cellule est un ensemble de j-cellules, $j < i$. Deux cellules sont dites *incidentes* si l'une appartient au bord de l'autre, et deux i-cellules sont dites *adjacentes* si elles sont incidentes à la même j-cellule, $j < i$. Les cartes combinatoires sont des structures qui décrivent toutes les subdivisions et les relations d'incidence entre toutes les différentes cellules de l'espace. Elles contiennent donc toute la topologie de cet espace.

La carte combinatoire de dimension deux représentant un objet 2D, s'obtient en décomposant les *faces* de l'objet puis les *arêtes* de ces faces dont les intersections sont les *sommets* de la carte. C'est l'étape de subdivision topologique (figure 5.18 page précédente (a)). Pour obtenir la carte formelle, il faut expliciter les relations topologiques à l'aide de *brins* qui sont les éléments de base d'une carte combinatoire (figure 5.18 page précédente (b)). Intuitivement, un brin représente « une moitié » d'arête, celle qui appartient à une des faces incidentes. Enfin la carte formelle se définit de la manière suivante :

Définition 23 (Cartes combinatoires de dimension deux) *Une carte combinatoire de dimension deux (ou 2-carte) est un triplet $M = (D, \beta_1, \beta_2)$ où :*

1. *D est un ensemble fini de brins ;*
2. *β_1 est une* permutation[2] *sur D.*
3. *β_2 est une* involution[3] *sur D.*

où β_1 est l'application qui associe à une arête la suivante sur la même face et β_2, l'application qui associe deux faces incidentes à la même arête. La figure 5.18 page précédente (c) représente la carte combinatoire représentant la subdivision topologique de la figure 5.18 page précédente (a). L'application β_1 (table 5.3 page précédente) est schématisée par l'orientation des brins et l'application β_2 est schématisée par la juxtaposition de deux brins d'orientation contraire. La composition de β_1 avec β_2 est notée $\beta_{21} = \beta_1 \circ \beta_2$, c'est une permutation. Soit $\Phi = \{f_1, \ldots, f_k\}$ un ensemble de permutations, le *groupe de permutations* généré par Φ, noté $\langle\Phi\rangle$, représente l'ensemble des permutations obtenues par toutes les compositions et inversions des permutations de Φ. L'*orbite* d'un brin est alors définie de la façon suivante :

Définition 24 (Orbite d'un brin) *Soit un brin d de l'ensemble D d'une carte combinatoire et $\langle\Phi\rangle$ un groupe de permutations définies sur D, l'*orbite *de d relativement à Φ, notée $\langle\Phi\rangle(d)$ est l'ensemble des brins de D tels que :*

$$\langle\Phi\rangle(d) = \{\phi(d) | \phi \in \langle\Phi\rangle\} \tag{5.8}$$

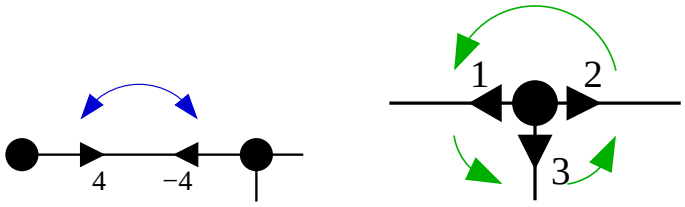
En dimension deux, trois orbites particulières permettent de retrouver les i-cellules. Ainsi le sommet incident à un brin d est défini par $\langle\beta_{21}\rangle(d)$[4], l'arête par $\langle\beta_2\rangle(d)$ et la face par $\langle\beta_1\rangle(d)$. Par exemple sur la figure 5.18 page précédente, le sommet incident au brin 1 est défini par $\langle\beta_{21}\rangle(1) = \{1, 14\}$, l'arête par $\langle\beta_2\rangle(1) = \{1, 13\}$ et la face par $\langle\beta_1\rangle(1) = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$.

BRUN ([Bru96]) utilise un autre formalisme basé sur une permutation σ équivalente à β_1 et une involution α équivalente à β_2 (figure 5.19 page suivante). Les arêtes sont superposées et représentées par deux flèches de sens opposés. Elles sont numérotées par un entier et son inverse ce qui permet d'avoir un codage implicite de l'involution α. Les sommets sont représentés par des points noirs dans ce formalisme.

[2] Une permutation sur D est une bijection de D dans lui-même.

[3] Une involution sur D est une permutation f telle que $f^{-1} = f$

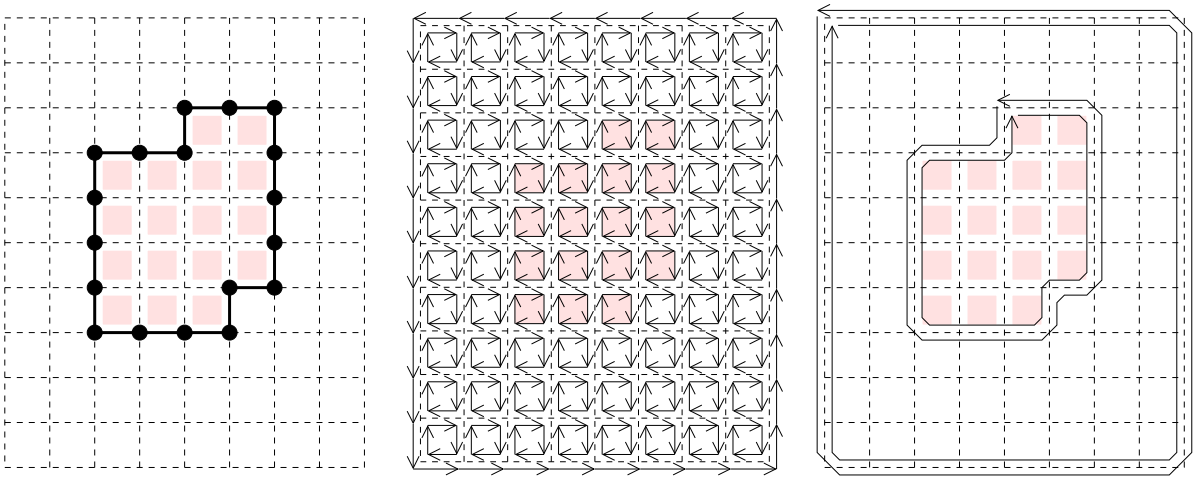
[4] Cette définition des sommets est valable uniquement si la carte est fermée, ce qui est le cas pour la représentation d'une image



(a) (b)

FIGURE 5.19 – Formalisme de BRUN ([Bru96]), (a) Involution α, (b) permutation σ.

Utilisation La manière la plus intuitive d'utiliser les cartes combinatoires pour la description d'une image est de considérer chaque région de l'image comme une face et d'associer les noeuds et arêtes aux points et segments constituant le contour interpixel des régions ([BB98]) (figure 5.20 (a)). DAMIAND ET COLL. ([DBF04]) proposent ainsi une description à plusieurs niveaux d'une image. Le niveau 0 est la carte combinatoire des pixels où chaque pixel correspond à une face (figure 5.20 (b)). Les niveaux supérieurs sont des cartes combinatoires de régions où les faces correspondent aux régions préalablement segmentées (figure 5.20 (c)). Le passage d'un niveau à l'autre se fait grâce à des opérations de simplification conservant la topologie de la carte. Des attributs géométriques ou de description peuvent également être associés aux brins.



(a) (b) (c)

FIGURE 5.20 – (a) Contour interpixel d'une région, (b) carte topologique de niveau 0, (c) carte topologique de niveau $k > 0$.

Nous avons essayé d'utiliser ces structures pour des images squelettisées de caractères hiéroglyphiques ([Leb05]). L'idée était d'associer, au niveau 0, un brin à chaque primitive du squelette (une primitive est définie par un morceau de squelette - strictement 4-connexe - compris entre deux points dont le nombre de voisins 4-connexes est différent de deux). Cette structuration est schématisée sur la figure 5.21 page suivante. L'objectif était de décrire ainsi toute une page de manuscrit hiéroglyphique et de réaliser l'extraction de la structure avec des opérations de simplification topologique. Malheureusement, la singularité de cette approche est telle qu'elle fait apparaître des arêtes pendantes (arête $(14, -14)$ par exemple sur la figure 5.21 page suivante (d)) et demande une redéfinition des opérations de simplification (la fusion des brins par exemple). De plus cette structuration n'apporte, dans ce cas, pas plus d'information qu'une pyramide de graphes.

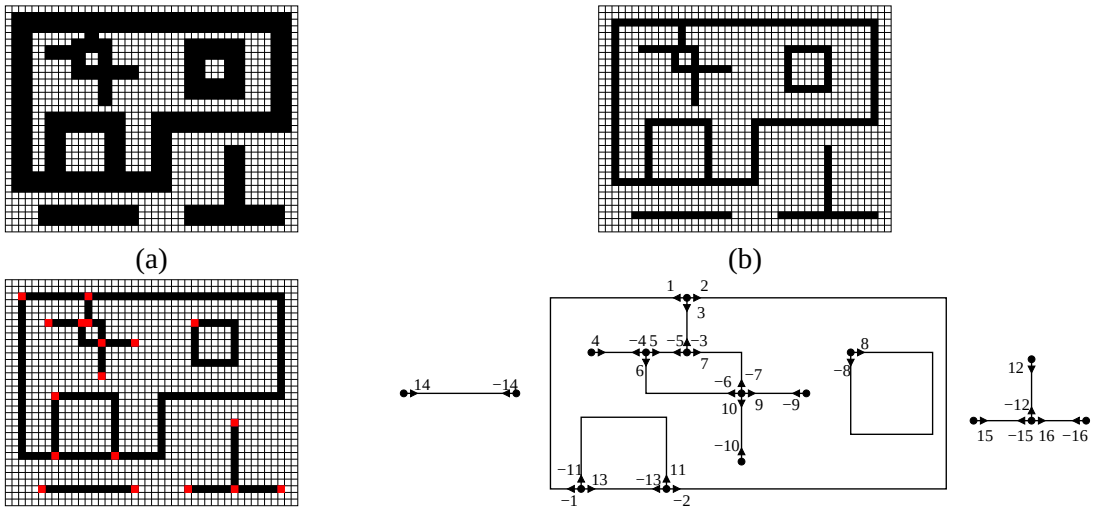


FIGURE 5.21 – Exemple de structuration de forme squelettisée par carte combinatoire ([Leb05]), (a) image d'origine, (b) squelette 4-connexe, (c) Subdivision topologique, (d) carte combinatoire avec le formalisme de BRUN ([Bru96]).

5.3.2.2 Les graphes

Les graphes sont des structures plus courantes mais qui ne sont pas topologiquement complètes.

Formalisme et Définitions Un *graphe* G est une paire $G = (N, A)$ d'ensembles satisfaisant $N \subseteq [A]^2$ avec $N \cap A^2 = \emptyset$. Les éléments de N sont appelés les *nœuds* ou *sommets* et les éléments de A les *arêtes* ou *arcs* de G.

L'*ordre* d'un graphe G est défini comme étant son nombre de nœuds et se note $|N|$. Le nombre d'arêtes se note $|A|$ et définit la *taille* du graphe.

Quand deux nœuds de G, notés u et v, sont connectés par une arête $a \in A$ ($a = (u, v)$), u et v sont dits *adjacents* ou *voisins*. Une arête connectant un nœud à lui même est appelée une *boucle*.

Si le couple (u, v) est ordonné, c.-à-d. ayant une *extrémité initiale* et une *extrémité finale*, l'arête $a = (u, v)$ est dite *orientée*. Un graphe G ayant toutes ses arêtes orientées est appelé un *graphe orienté*[5]. Un *multigraphe* est un graphe pour lequel il peut exister plusieurs arêtes entre deux sommets. Un graphe est dit *simple* si :

– il est sans boucle,
– il n'y a jamais plus d'une arête entre deux sommets quelconques.

Si tous les noeuds de G sont deux à deux adjacents, G est dit *complet*. Un graphe simple complet est souvent noté K^n avec $n = |N|$.

Les sommets et les nœuds peuvent contenir des informations. Quand l'information est une étiquette (c.-à-d. un nom ou un numéro), le graphe est appelé un *graphe étiqueté*. Quand l'information est plus complète et contient des attributs (géométriques, de couleurs...), le graphe est appelé *graphe d'attributs*.

[5]Dans le cas d'un graphe orienté, le terme d'arc est préféré à celui d'arête.

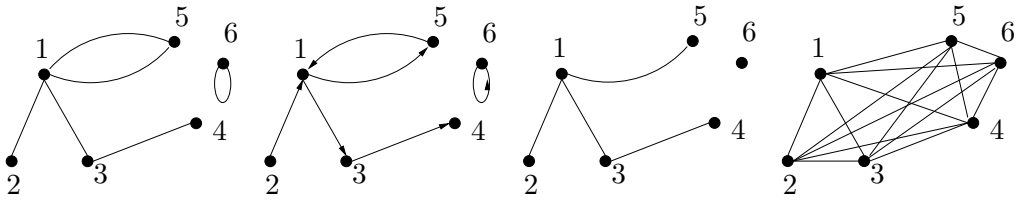


FIGURE 5.22 – Exemples de graphes avec $N = \{1, \ldots, 6\}$.

Un *chemin* entre deux sommets $u, u' \in N$ est une séquence non vide de k sommets $(v_0, v_1, \ldots, v_k)$ telle que $u = v_0$, $u' = v_k$ et $(v_{i-1}, v_i) \in A$, $i = 1, \ldots, k$. Une *chaîne* est un chemin pour lequel l'orientation éventuelle des arcs n'est pas prise en compte. Un *cycle* est une chaîne dont les extrémités coïncident. Le graphe G est dit *acyclique* s'il n'existe aucun cycle entre ses sommets. Les graphes acycliques sont aussi appelés *forêt* et une forêt connectée (c.-à-d. pour laquelle il existe toujours une chaîne entre deux sommets) est un *arbre*.

Utilisation Les graphes d'adjacence de régions (GAR) sont très utilisés pour modéliser une image ([Ros74]). Les nœuds décrivent les différentes régions de l'image segmentée et les arêtes décrivent les relations d'adjacence entre régions. Ils peuvent être également intégrés dans une représentation multi-échelles de l'image permettant la recherche d'un objet à différents niveaux de résolution comme dans les travaux de DOMBRE ([Dom03]).

En reconnaissance de formes dans une scène simple (ne contenant que la forme à reconnaître), le graphe d'adjacence de régions n'a que peu d'intérêt puisque c'est la région elle-même que l'on cherche à reconnaître. Les structures de graphes sont alors basées sur le contour ou sur le squelette. Les plus connus sont les graphes de chocs (*shock graphs*, [SSDZ99]) qui sont des graphes orientés étiquetés. Ils sont basés sur un calcul de l'axe médian par évolution de courbe. Le contour est paramétré à l'aide d'une interpolation ENO (« Essentially Non-Oscillatory ») permettant de tenir compte des discontinuités ([SKS95]). Puis l'évolution intérieure suivant une équation aux dérivées partielles donne un squelette étiqueté ($1, 2, 3$ ou 4) suivant la dérivée en ce point de la fonction rayon (figure 5.23 page suivante). Le graphe est construit à partir du squelette, chaque groupe de pixels adjacents de même étiquette est représenté par un nœud et les arcs relient les goupes adjacents. Les graphes ainsi obtenus sont orientés, acycliques et connectés, donc équivalents à des arbres orientés. SEBASTIAN ET COLL. ([SKK04]) proposent une variante des graphes de chocs décrivant mieux la topologie de la forme et basée sur le calcul d'axe médian par la propagation Eulérienne-Lagrangienne de TEK ET COLL. ([TK03]).

En reconnaissance de caractères, l'information de forme contenue dans les graphes de chocs n'a, *a priori*, que peu d'intérêt. En effet les formes sont bien souvent minces et seul le squelette suffit à leur interprétation. En écriture latine, le squelette d'un « A » n'est finalement que la même lettre écrite plus finement. C'est pourquoi la majorité des descriptions par graphes utilisées en reconnaissance de caractères est basée sur le squelette lui-même. Celui-ci est découpé en segments primitifs représentés par les nœuds, les arêtes expriment l'adjacence de ces segments. Afin de compléter la description, des attributs géométriques sont souvent associés aux nœuds et aux arêtes. Les travaux les plus connus sont ceux de KIM ET KIM ([KK01]). Ils utilisent une description à trois niveaux pour la reconnaissance des caractères coréens. Les graphes d'attributs sur les segments primitifs constituent le niveau le plus bas. Au niveau supérieur, les nœuds sont associés à des graphèmes (ensemble de segments primitifs) et les arêtes décrivent leurs relations d'adjacence. Enfin au dernier niveau, les nœuds représentent les composantes connexes des caractères (les caractères coréens, comme les caractères chinois, sont composés de plusieurs composantes connexes) et leurs relations spatiales. La figure 5.24 page suivante présente la structure à trois

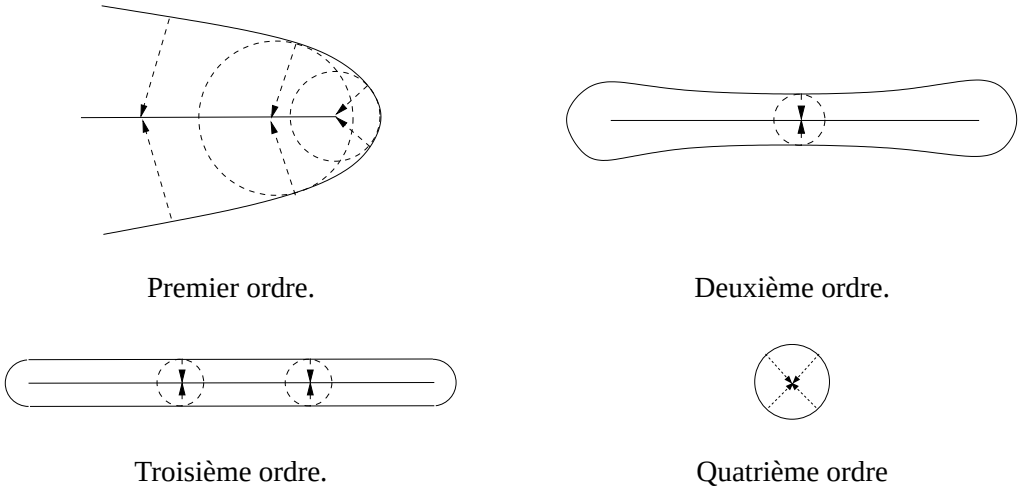


FIGURE 5.23 – Etiquettes des graphes de chocs.

niveaux, les graphes appelés « Random Graphs » sont une généralisation des graphes d'attributs servant à modéliser une base d'apprentissage, nous reviendrons sur ce concept dans la partie reconnaissance.

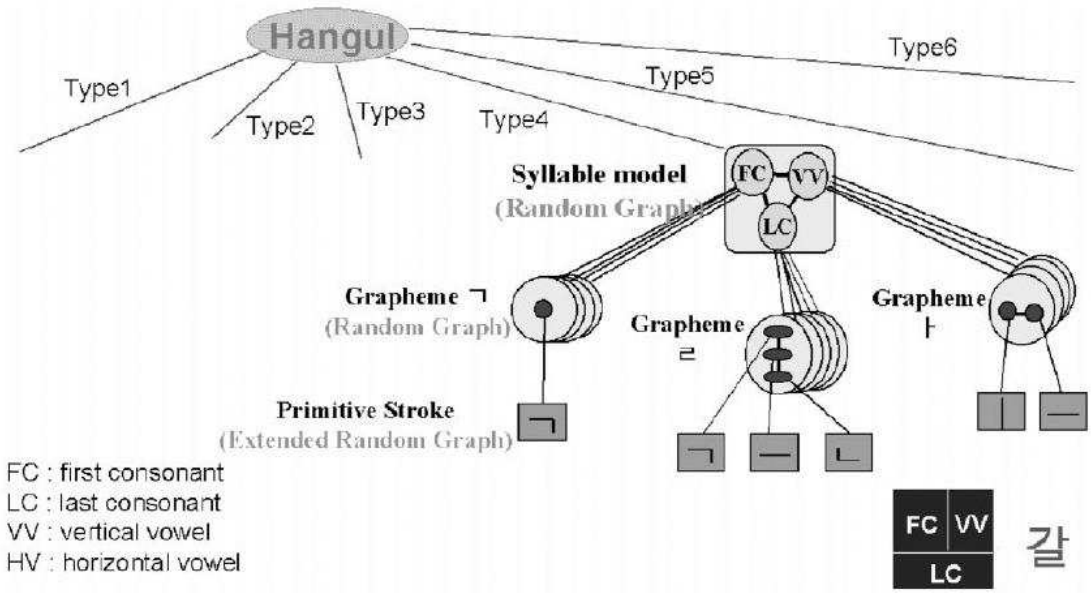


FIGURE 5.24 – La description hiérarchique de KIM ET KIM ([KK01]).

Citons également les travaux de CHAN ET CHEUNG ([CC92]) et de MAN ET POON ([MP93]) qui utilisent des graphes d'attributs flous sur lesquels nous reviendrons.

5.4 Synthèse

La manière la plus naturelle d'exprimer la structure d'un caractère est d'en extraire le squelette. Parmi les méthodes de squelettisation et dans le cadre de notre étude, l'algorithme de ZHANG ET WANG

est un bon candidat, paragraphe 5.1.3.5 page 70. A partir de ce squelette, l'extraction des points d'intersection et de fin se fait naturellement. Nous avons ensuite présenté une technique de paramétrage des segments singuliers permettant d'extraire les points d'inflexion en s'affranchissant relativement bien du bruit. Enfin, en considérant les différentes structures existant dans la littérature, les graphes d'attributs semblent relativement bien appropriés pour modéliser nos caractères.

La structuration se fait donc en trois étapes (sans les prétraitements qui sont spécifiques à chaque base et explicités en Annexe A page 155) : la squelettisation à partir de la méthode de ZHANG ET WANG, la détection des points singuliers (points de fin, d'intersection et d'inflexion) et la construction du graphe d'attributs. Nous avons utilisé, pour les comparer, deux types de graphes. Pour les premiers, les nœuds représentent les points singuliers et les arêtes les segments primitifs. Nous les appelerons des *graphes primaires*. Pour les seconds, les nœuds représentent les segments primitifs et les arêtes leurs relations d'adjacences, il s'agit des graphes d'adjacence classique que nous nommerons *graphes d'arêtes*. Cette double structuration est schématisée sur la figure 5.25.

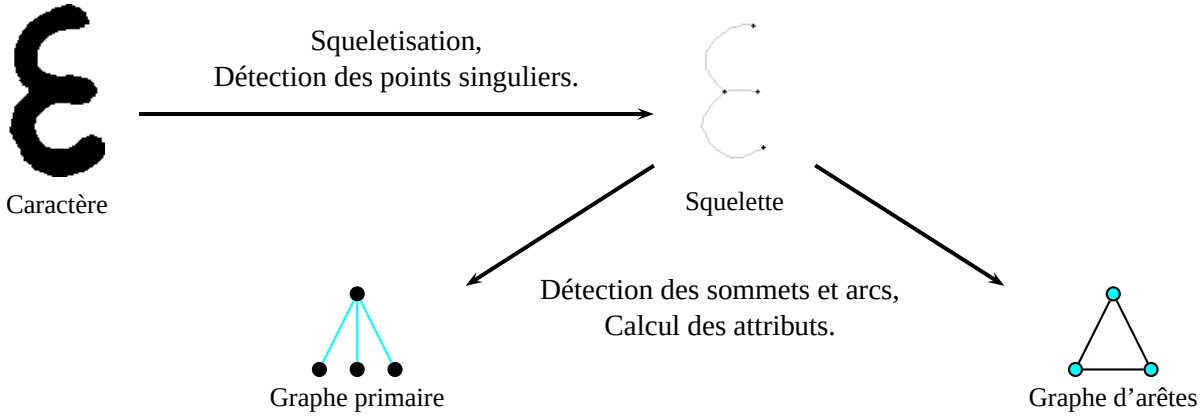


FIGURE 5.25 – Chaîne de construction d'un graphe à partir d'un caractère

Le choix des attributs associés aux nœuds et aux arêtes est dépendant de la structure choisie et influence le système de reconnaissance. Notons enfin que les graphes que nous avons appelés primaires sont des multigraphes non-orientés pouvant posséder des boucles alors que les graphes que nous avons appelés d'arêtes sont des multigraphes non orientés sans boucle.

Comment utiliser cette structuration de l'information dans un système de reconnaissance avec apprentissage ? Le prochain chapitre propose un schéma de mise en œuvre basé sur des techniques d'appariement de graphes.

CHAPITRE 6

DESCRIPTION D'UN SYSTÈME PROBABILISTE DE RECONNAISSANCE STRUCTURELLE

Sommaire

La reconnaissance de structures telles que celles définies précédemment est un problème assez délicat qui devient vite très coûteux en temps de calcul. Par rapport aux méthodes statistiques numériques, la reconnaissance se fait maintenant à deux niveaux. Il faut d'une part mesurer la similarité des structures et d'autre part mesurer la similarité des composantes primitives et ce bien souvent avec des formalismes différents. De nombreuses méthodes peuvent être dérivées suivant le type de structures utilisées et la nature des caractéristiques associées aux primitives. Les deux principales familles sont les approches *syntaxiques* ([Bri99]) qui se dérivent de manière *stochastique* avec les modèles markoviens et les approches à base de mise en correspondance de graphes ou approches *graphiques* ([Bun00]).

Les approches syntaxiques reposent sur la *théorie des langages* dont les modèles remontent aux années 50 et les développements de CHOMSKY ([Cho56]). Les formes sont codées par des mots utilisant un *alphabet* dont chaque terme représente un élément de la forme à décrire. Chaque classe de formes est alors définie par un ensemble de règles syntaxiques, ou *grammaire*, définissant les mots acceptables. Le choix de l'alphabet et de la grammaire, qui, ensemble, forment un *langage*, incombe à des méthodes d'*inférence* relativement délicates à mettre en oeuvre. La reconnaissance formelle d'une forme à partir d'un langage se fait ensuite à l'aide d'*automates*. FU est un des précurseurs de l'emploi de ce formalisme en reconnaissance de formes ([Fu82]). Le formalisme purement déterministe des langages syntaxiques étant peu aproprié à la reconnaissance de formes, les règles ont évolué vers des formes floues et probabilistes pour donner naissance aux grammaires floues et stochastiques ([dlH05] publié récemment dans un numéro spécial sur l'inférence grammaticale de la revue Pattern Recognition).

Les approches *stochastiques* reposent le plus souvent sur des modèles markoviens. Suivant la description, trois modèles sont adaptables, les modèles $1D$ ou *chaînes de Markov*, les modèles pseudo-$2D$ ou *modèles planaires* et les modèles $2D$ ou *champs de Markov* ([BS97]). Chaque élément de la description structurelle est un site associé à une variable aléatoire. Le caractère, ou champ des observables, est la réalisation du vecteur aléatoire réunissant chaque variable de site. L'objectif est d'estimer le champ des étiquettes associées à chaque site à partir du champ des observables. Le cadre markovien permet l'utilisation d'une notion de voisinage entre sites pour estimer la probabilité des champs aléatoires.

Les approches *graphiques* reposent, elles, sur des techniques *combinatoires* pour mesurer la ressemblance entre graphes. Là encore, pour donner plus de souplesse aux modèles et permettre la reconnaissance d'informations bruitées, des approches floues ([PB02]) et bayésiennes ([CKP95]) ont été dérivées.

Utilisant des graphes d'attributs, notre schéma de reconnaissance se place dans la catégorie des méthodes graphiques. Il repose sur un apprentissage et une reconnaissance basés sur l'appariement de graphes. C'est ce que nous expliquerons dans la première partie de ce chapitre. Nous décrirons dans une seconde partie la méthode de construction des graphes d'attributs utilisés. Ensuite nous aborderons la problématique de l'appariement de graphes et détaillerons les algorithmes que nous avons mis en place dans notre phase d'apprentissage. Enfin nous expliquerons notre système de reconnaissance probabiliste.

6.1 Positionnement du problème

Le problème de la reconnaissance graphique par appariement de graphes, comme nous le verrons dans la suite de ce chapitre, est un problème relativement complexe qui nécessite une définition précise du cadre de recherche. Il y a, comme pour la reconnaissance statistique, deux étapes à différencier qui sont l'apprentissage ou classification supervisée et la reconnaissance à proprement parler. Alors que l'apprentissage va consister à schématiser l'ensemble des éléments d'une classe par une structure « moyenne » munie d'un ensemble d'attributs résumant ceux des graphes de la classe, la reconnaissance revient à mesurer une similarité entre un graphe et les éléments « moyens » de la phase précédente. Quand la base est peu fournie, la reconnaissance peut consister à mesurer la similarité du graphe inconnu avec tous les éléments de la base puis à utiliser une règle des k plus proches voisins pour se replacer dans un cadre bayésien classique. Malheureusement les appariements de graphes sont souvent couteux en temps de calcul et cette approche devient vite prohibitive.

Notre étude s'intègre dans le cadre de la reconnaissance générique de caractères. Même si ce cadre est ambitieux, il n'en reste pas moins que nous ne pouvons pas, *a priori*, avoir de connaissance sur le type de caractères à reconnaître, nous n'avons pas de modèle au sens strict. En revanche, nous possédons une base de caractères connus plus ou moins proches de ceux que nous voulons reconnaître. Trouver une structure « représentative » pour une classe à partir de la base d'apprentissage n'est donc pas un gage

de bonne reconnaissance mais à l'inverse, comme nous l'avons dit précédemment, conserver tous les graphes appris pour la reconnaissance n'est pas une solution.

La recherche de graphes « représentatifs » a été abordée par JIANG ET COLL. ([JMB01]) qui proposent le graphe médian généralisé d'un ensemble de graphes. Il s'agit du graphe le plus proche de tous les autres graphes de la classe au sens d'une fonction de distance inter-graphe. La distance structurelle proposée est une distance d'édition basée sur une fonction de coût de transformation d'un graphe à un autre à partir de six opérations : la supression, l'insertion et la substitution ceci appliqué sur les nœuds et les arêtes. BUNKE ([Bun97]) montre par ailleurs que sous certaines conditions sur la fonction de coût, la distance d'édition est directement liée à la cardinalité du *sous-graphe commun maximum*.

Cependant, la distance d'édition est, dans la définition de JIANG et BUNKE ([Bun97], [JMB01]), basée uniquement sur la structure. Pour prendre en compte la nature des attributs il faut redéfinir la fonction de coût à partir d'une mesure de similarité entre attributs. La relation établie avec la cardinalité du sous-graphe commun maximum n'est alors plus valable.

Avec des graphes d'attributs, le graphe médian peut être vu comme le graphe le plus proche de tous les éléments de la classe au sens d'une métrique basée sur un appariement. Dans ce cas la mesure de similarité englobe les erreurs issues de l'appariement. La recherche du graphe médian, en plus d'être extrêment coûteuse puisqu'elle suppose l'appariement deux à deux de tous les graphes de la base, introduit une imprécision qui, cumulée avec celle de la reconnaissance, peut être préjudiciable au système.

Comment alors justifier le choix d'une structure « représentative » pour chaque classe ? Si nous partons de l'hypothèse restrictive consistant à présupposer que les bases d'apprentissage ne sont constituées que de caractères bien segmentés et bien étiquetés, les variations intra-classe ne sont alors dues qu'aux différences de scripteurs et de périodes. Qu'est-ce qui fait, dans ce cas, que nous pouvons associer la même étiquette à des caractères aussi différents que ceux de la figure 6.1 ? Cet exemple montre que la notion de structure médiane n'a pas beaucoup de sens dans notre cas. Nous devons accorder plus d'importance à l'appariement de primitives semblables qu'à la cohérence structurelle de la classe. Qui plus est, cet exemple met en évidence la limite des approches structurelles qui ne peuvent pas apporter une réponse unique au problème de la reconnaissance. C'est pourquoi il est, *a priori*, préférable de les utiliser en complément d'une reconnaissance statistique. Nous ne pouvons pas admettre non plus un double étiquetage de la classe basé sur un critère de rejet. Ce serait prendre le risque d'avoir autant de sous-classes que de scripteurs ou au moins de périodes historiques (impliquant une différence de structuration des lettres). Le surcoût calculatoire pour la phase de reconnaissance ne serait pas négligeable.

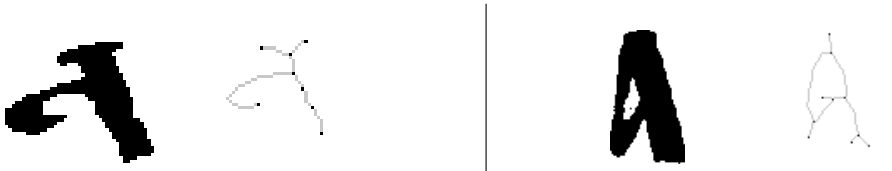


FIGURE 6.1 – Exemples de caractères λ tirés de la base GrAnc (Annexe A page 155) avec les structures associées

Partant de ces constatations, nous avons choisi de conserver comme modèle de classe, la structure la moins contraignante. Or, en terme d'appariement, la structure la moins contraignante est celle qui contient le moins de nœuds. Le modèle d'une classe sera donc le graphe de plus petit ordre avec un ajustement possible par retour d'expertise en cas de mauvaise classification.

A partir de ces modèles les deux étapes du système de reconnaissance se décomposent ainsi. En premier lieu, l'apprentissage (figure 6.2) va consister à associer à chaque primitive et relation du modèle un ensemble d'observations provenant de la base d'apprentissage. Pour ce faire, nous utiliserons des techniques d'appariement approximatif de graphes. En second lieu, à partir des observations sur chaque élément du graphe modèle, un modèle probabiliste est construit qui permet de formaliser la reconnaissance.

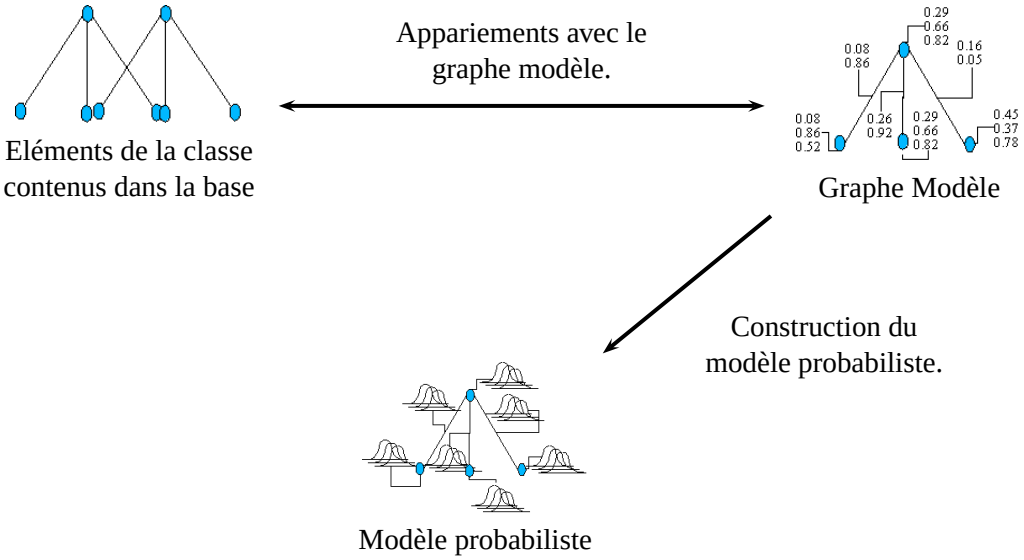


FIGURE 6.2 – Principe de l'apprentissage

6.2 Les graphes d'attributs

6.2.1 Définition

Les graphes d'attributs ont été introduits en reconnaissance de formes par TSAI ET FU ([TF83]). Par définition, les nœuds des graphes représentent les primitives tandis que les arêtes expriment leurs relations. Dans le cas des graphes que nous avons appelés primaires, les primitives sont donc les points singuliers du squelette et dans le cas des graphes que nous avons appelés d'arêtes, il s'agira des segments primitifs. Chaque nœud du graphe prend ses attributs dans l'ensemble $Z = \{z_i | i = 1, ..., I\}$ où chaque attribut z_i va prendre ses valeurs dans l'ensemble $S_i = \{s_{ij} | j = 1, ..., J_i\}$. L'ensemble $L_v = \{(z_i, s_{ij}) | i = 1, ..., I; j = 1, ..., J_i\}$ est l'ensemble des couples de valeurs possibles pour les primitives. Une primitive valide est alors un sous-ensemble de L_v dans lequel chaque attribut ne peut apparaître qu'une seule fois. Si Π est l'ensemble de toutes ces primitives valides, chaque nœud est représenté par un élément de Π.

De manière similaire, pour les arêtes, l'ensemble des attributs possibles est appelé $F = \{f_i | i = 1, ..., I'\}$ dans lequel chaque f_i peut prendre ses valeurs dans $T_i = \{t_{ij} | j = 1, ..., J'_i\}$. L'ensemble $L_a = \{(f_i, t_{ij}) | i = 1, ..., I'; j = 1, ..., J'_i\}$ est l'ensemble des couples de valeurs possibles pour les relations. Une relation valide est alors un sous-ensemble de L_a dans lequel chaque attribut ne peut apparaître qu'une seule fois. L'ensemble de toutes les relations valides est noté Θ.

Un graphe d'attributs se définit formellement de la manière suivante :

Définition 25 (Graphe d'attributs ou GA) *un graphe d'attribut Ga sur $L = \{L_v, L_a\}$ avec une structure graphique $G = (N, A)$, est une paire ordonnée (V, E) où $V = (N, \sigma)$ est appelé un* ensemble de nœuds

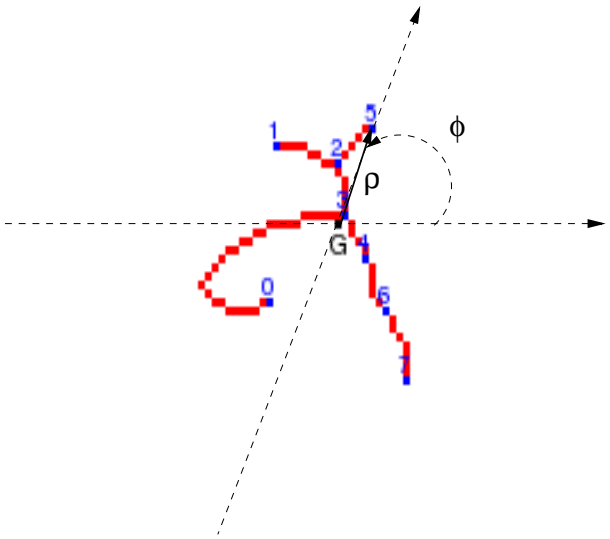


FIGURE 6.3 – Attributs géométriques liés aux nœuds des graphes primaires - nœud 5 (ρ, ϕ).

attribués *et* $E = (A, \delta)$ *est appelé un* ensemble d'arêtes attribuées. *Les applications* $\sigma : N \rightarrow \Pi$ *et* $\delta : A \rightarrow \Theta$ *sont appelées respectivement* interpréteurs *de nœuds et d'arêtes.*

Nous pouvons, dans notre cas, introduire deux simplifications de notation. Tout d'abord, comme nous allons le voir par la suite, nos attributs sont continus à valeurs réelles sur $[0, 1]$ ce qui implique que $S_i = T_i = [0, 1]$. Ensuite, il est souvent intéressant d'exprimer toutes les valeurs des attributs d'un même élément par un vecteur. Nous noterons $\boldsymbol{\alpha}_a$ le vecteur des valeurs des attributs du nœud a et $\boldsymbol{\beta}_{ab}$ le vecteur des valeurs des attributs de l'arête (a, b).

6.2.2 Construction

Les attributs attachés aux éléments du graphe décrivent la géométrie du caractère. L'information modélisable dépend donc fortement du type de graphe considéré.

6.2.2.1 Graphes primaires

Dans les structures primairees, les relations d'adjacence entre segments sont définies dans la structure même. Les informations géométriques contenues dans les nœuds se résument alors à leur positionnement spatial, les arêtes correspondant aux informations de forme des segments primitifs.

Les attributs retenus pour les nœuds sont les coordonnées polaires par rapport à un repère intrinsèque au caractère (notés ρ et ϕ, voir figure 6.3). Nous avons pris le centre de gravité du squelette comme centre du repère. Pour l'axe de référence angulaire, le choix s'est porté de façon empirique sur l'horizontale de l'image du caractère plutôt que sur l'axe principal ou de moindres carrés. L'erreur générée par ces derniers étant beaucoup plus dommageable à la reconnaissance que la perte de l'invariance par rotation qu'implique l'utilisation de l'axe horizontal. Le rayon et l'angle polaire sont normalisés respectivement par le rayon maximal sur l'image et 2π afin d'obtenir deux attributs à valeurs dans $[0, 1]$.

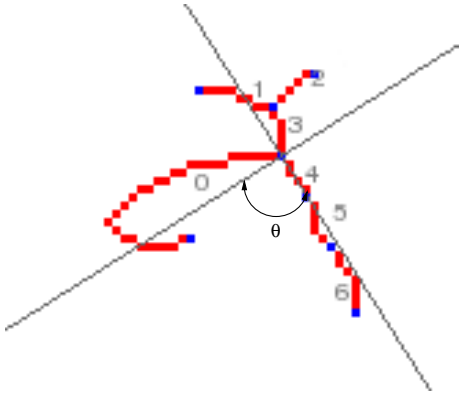


FIGURE 6.4 – Attribut géométrique lié aux arêtes des graphes d'arêtes - relation entre les nœuds 0 et 4 (θ).

Pour les relations, les attributs décrivent la longueur relative, lr, du segment et sa rectitude (« straightness » [PDM86]), St, données pour l'arête reliant les nœuds a et b par :

$$lr_{ab} = \frac{\overset{\frown}{ab}}{\sum_{(c,d)\in A} \overset{\frown}{cd}} \tag{6.1}$$

où $\overset{\frown}{ab}$ représente la longueur d'arc de l'arête (a, b) et le dénominateur la somme des longueurs d'arc de toutes les arêtes du graphe.

$$St_{ab} = \frac{ab}{\overset{\frown}{ab}} \tag{6.2}$$

où ab désigne la longueur du segment droit reliant le point a et le point b. La longueur relative comme la rectitude sont à valeurs dans $[0, 1]$. Ainsi, un nœud a est associé à un vecteur d'attributs $\boldsymbol{\alpha}_a = (\rho, \phi)^\perp$ et une arête (a, b) au vecteur $\boldsymbol{\beta}_{ab} = (lr_{ab}, St_{ab})^\perp$.

6.2.2.2 Graphes d'arêtes

Dans les graphes d'arêtes, les nœuds représentent les relations des graphes primaires et gardent donc les mêmes attributs à savoir lr et St. La notation, elle, change et $\boldsymbol{\alpha}_a = (lr_a, St_a)^\perp$. Les relations d'adjacence sont données par la présence ou l'abscence d'arête entre sommets, un attribut de positionnement angulaire relatif est associé à chaque arête. Le positionnement angulaire relatif (noté θ, figure 6.4) est établi par l'angle que forment les deux axes principaux des segments. Cet angle est non orienté et normalisé par π afin que l'attribut soit à valeur dans $[0, 1]$.

L'axe principal d'un segment primitif correspond au vecteur propre principal de l'ensemble de ses points. Ainsi pour une arête (a, b), $\boldsymbol{\beta}_{ab} = (\theta_{ab})$.

6.2.3 Similarités entre éléments

Pour réaliser un appariement de deux graphes, il nous faut définir une mesure de comparaison entre les nœuds et les arêtes prenant en compte la ressemblance des attributs. D'une manière générale, toute similarité définie dans un cadre statistique entre vecteurs numériques est une candidate valable. Comme

tous les attributs sont à valeurs dans $[0,1]$ et ont la même importance, une similarité basée sur une distance de Minkowski (2.62 page 32) sans pondération est suffisante :

$$s(\mathbf{x},\mathbf{y}) = 1 - \frac{d_{Mink}(\mathbf{x},\mathbf{y})}{n^{\frac{1}{p}}} \tag{6.3}$$

où $\mathbf{x}$ et $\mathbf{y}$ représentent les vecteurs des valeurs des attributs à comparer ($\boldsymbol{\alpha}_i$ ou $\boldsymbol{\beta}_{ij}$), $n = I$ ou I' selon le type d'élément considéré et p est l'ordre de la distance choisie. Nous avons pris la distance de Minkowsky d'ordre 1 car dans ce cas précis les ordres supérieurs n'apportent pas plus d'information. Nous noterons $s_N(a,b)$ la similarité entre deux nœuds a et b et $s_A(a,b;i,j)$ la similarité entre les arêtes (a,b) et (i,j).

Il reste néanmoins une petite difficulté à régler qui est celle de la comparaison d'attributs périodiques comme c'est le cas avec les mesures angulaires. La soustraction est alors remplacée par une différence périodique définie de la façon suivante :

$$\Delta(x,y) = \begin{cases} x-y & \text{si } |x-y| \leq \frac{T}{2} \\ x-y-T\times \text{sign}(x-y) & \text{si } |x-y| > \frac{T}{2} \end{cases} \tag{6.4}$$

où (x,y) sont deux réels périodiques de période T et sign$(x-y)$ le signe de leur différence. Dans notre cas $T=1$.

6.3 Apprentissage par appariement de graphes

L'appariement de graphes focalise l'attention de beaucoup d'activités de recherche de la communauté depuis plus de 20 ans ([Ull76], [KO90], [PCB94]). Il existe plusieurs approches suivant le type d'appariement recherché (exact ou inexact) et les types de graphes à apparier (étiquetés, valués, cycliques...).

6.3.1 Appariement exact et inexact de graphes

Un appariement entre deux graphes de même dimension est un isomorphisme de graphe qui peut se définir de la façon suivante :

Définition 26 (Isomorphisme de graphe) *Soit deux graphes non orientés non attribués $G_1(N_1,A_1)$ et $G_2(N_2,A_2)$, tels que $|N_1| = |N_2|$, un isomorphisme de (G_1,G_2) est une application bijective $f : N_1 \rightarrow N_2$ telle que $(u,v) \in A_1$ si et seulement si $(f(u),f(v)) \in A_2$*

G_1 et G_2 sont alors dits *isomorphiques*. La détermination d'un isomorphisme de graphe correspond au problème d'*appariement exact de graphes*. Si la dimension des graphes est différente, il est toujours possible de trouver un ou plusieurs isomorphismes entre des sous-graphes de G_1 et G_2. Il s'agit du problème d'*appariement exact de sous-graphes*. La complexité de l'appariement exact de graphe ou de sous-graphe, bien que non encore établie comme NP ([Kar72]), est importante. Dans le cas des graphes d'attributs, il faudra tenir compte de la similarité des nœuds et arêtes à apparier. Une application bijective n'est alors pas toujours le meilleur appariement, et il peut être préférable de résoudre un problème d'*appariement inexact de graphes*. La notion d'isomorphisme est élargie à celle d'*homomorphisme*.

Définition 27 (Homomorphisme de graphe ou morphisme) *Soit deux graphes $G_1(N_1,A_1)$ et $G_2(N_2,A_2)$, un homomorphisme de (G_1,G_2) est une application $f : N_1 \rightarrow N_2$ qui conserve les propriétés d'adjacence et de non-adjacence.*

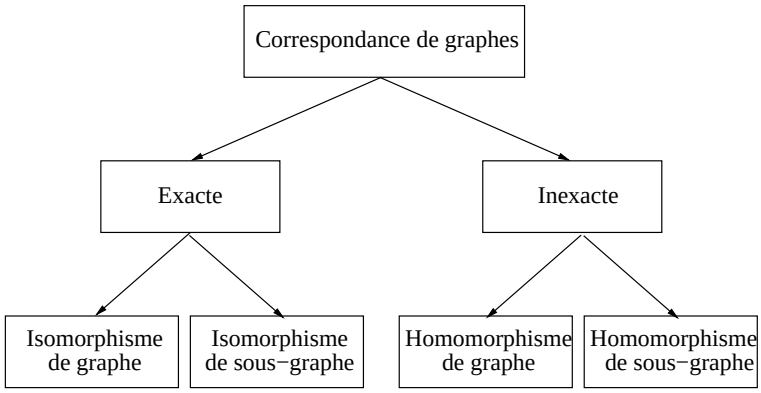


FIGURE 6.5 – Les différents problèmes d'appariement de graphes.

Dans le cas des modélisations de squelettes, la conservation de l'adjacence par un homomorphisme ou un isomorphisme peut se révéler contraignante. Il faut alors déterminer des applications qui dérogent « un peu » à cette règle. Ce sont des homomorphismes ou isomorphismes approximatifs que nous appellerons tout de même homomorphismes ou isomorphismes par commodité. La figure 6.5 présente les différents domaines de l'appariement de graphes.

Dans notre cas, par convention, le graphe de départ sera le graphe modèle. L'objectif est alors d'apparier le maximum de nœuds et d'arêtes du graphe modèle avec celui d'arrivée. Nous ajouterons deux règles à cet appariement. Premièrement, si nous partons de l'hypothèse que le graphe modèle est la structure minimum d'une classe, il n'est donc pas redondant. L'homomorphisme cherché ne peut pas être surjectif. Deuxièmement, le graphe d'arrivée est considéré comme bruité ce qui peut rompre les relations d'adjacence. L'homomorphisme cherché est approximatif. La première hypothèse nous place dans le cadre formel des isomorphismes de sous-graphes dont la complexité pourra être réduite grâce à la seconde hypothèse.

6.3.2 Méthodes globales d'appariement

La recherche d'un isomorphisme de sous-graphe entre deux graphes d'attributs a été largement traitée dans la littérature. Notre exposé ne sera donc pas exhaustif et nous essayerons surtout de bien formuler le problème et d'introduire des algorithmes efficaces pour le résoudre. Il y a deux façons d'aborder le problème dont dérivent deux familles de méthodes dites *globales* et *locales*.

Les premières approches globales utilisent un espace d'états explicite qui est parcouru exhaustivement afin de trouver la meilleure solution. ESHERA ET FU ([EF84]) proposent ainsi un appariement en trois phases consistant, premièrement, à décomposer les deux graphes en graphes basiques qui sont utilisés, dans la seconde phase, pour construire un espace d'états exprimant toutes les possibilités de constructions parallèles des deux graphes. La troisième phase consiste alors à chercher le chemin le plus court dans cet espace d'états qui correspondra à la distance d'édition la plus faible. TSAI ET FU ([TF79]) utilisaient une méthode similaire basée sur le maximum de vraisemblance de transformation d'un graphe en l'autre mais qui ne traitait que le cas des isomorphismes de graphes. D'une manière générale, ces méthodes permettent de formuler le problème de manière complète et de trouver une solution globale. En revanche, elles sont extrêmement coûteuses en temps de calcul et rédhibitoires pour le traitement de graphes de dimensions élevées. L'algorithme proposé par ESHERA ET FU, par exemple, est, dans le pire des cas, de complexité $\mathcal{O}\left(|N|^2|A|^2(|M|+|A|)\right)$.

Une autre approche globale, similaire en complexité combinatoire, consiste à utiliser un graphe d'association ([YSB89], [PSZ98]). Nous présentons ici les principes d'un appariement basé sur cette tech-

nique dans le cas des isomorphismes de graphes et de sous-graphes puis dans sa version étendue aux graphes d'attributs.

6.3.2.1 Construction du graphe d'association

Les graphes d'association ont été introduits par BARROW ET BURSTALL ([BB76]) pour résoudre les problèmes de recherche d'isomorphismes de graphes et de sous-graphes.

Définition 28 (Graphe d'association) *Considérons deux graphes non attribués $G_1 = (N_1, A_1)$ et $G_2 = (N_2, A_2)$, le* graphe d'association *de G_1 et G_2 est un graphe non-orienté $G = (N, A)$ défini par :*

$$\begin{gathered} N = N_1 \times N_2 \\ A = \{((i,h),(j,k)) \in N \times N \text{ tels que } i \neq j,\ h \neq k,\ \text{et } (i,j) \in A_1 \Leftrightarrow (h,k) \in A_2\} \end{gathered} \tag{6.5}$$

Définition 29 (Clique) *Soit $G = (N, A)$ un graphe non orienté, un sous-ensemble C de N est appelé une* clique *si tous les nœuds de C sont mutuellement adjacents, c.-à-d., $\forall i, j \in C,\ (i,j) \in A$.*

Une clique est dite *maximale* si elle n'est incluse dans aucune clique plus large. Une clique est dite *maximum* s'il s'agit de la clique de plus grande cardinalité sur G. La relation entre la clique maximum et le problème de recherche d'un isomorphisme de graphe est donné par le théorème suivant :

Théorème 4 (Isomorphisme de graphe) *Si $G = (N, A)$ est le graphe d'association de $G_1 = (N_1, A_1)$ et $G_2 = (N_2, A_2)$, deux graphes d'ordre n, alors G_1 et G_2 sont isomorphiques si et seulement si la cardinalité de la clique maximum est égale à n.*

La preuve est évidente, supposons que ϕ soit un isomorphisme de G_1 sur G_2, alors la clique $C_\phi = \{(i, \phi(i)) | \forall i \in N_1\}$ est clairement une clique maximum. Et inversement si C est une clique maximum de G d'ordre n, l'application $\phi : N_1 \rightarrow N_2$ définie par $\forall (i,h) \in C, \phi(i) = h$, est un isomorphisme de G_1 sur G_2.

De la même manière une clique maximale de cardinalité quelconque sur le graphe d'appariement définit un isomorphisme de sous-graphe dont l'ordre est égal à la cardinalité de la clique.

6.3.2.2 Recherche de cliques maximales

La recherche des cliques maximales étant de complexité NP, la solution ne peut être obtenue que par un parcours exhaustif de l'ensemble des chemins du graphe. Il existe malgré tout une littérature très fournie sur le sujet proposant des heuristiques pour réduire la complexité en fonction des types de graphes considérés ([AM70], [BB76], [BBPP99]). Une méthode exhaustive classique en complexité $\mathcal{O}(2^n)$ est présentée dans [RC92], n est le nombre de nœuds du graphe. Elle présente l'avantage d'être facilement parallélisable :

1. Initialisation : mettre tous les nœuds dans la même clique.
2. Parcourir l'ensemble des cliques en regardant si les nœuds sont deux à deux adjacents. S'ils ne le sont pas, la clique est supprimée et deux nouvelles cliques sont créées. Les deux nouvelles cliques contiennent chacune un des deux nœuds non adjacents et tous les autres nœuds de la clique supprimée. Elles sont ajoutées à la fin de l'ensemble des cliques.
3. Réitérer l'étape précédente sur chaque clique potentielle. Quand plus aucune division n'est possible, l'ensemble des cliques contient toutes les cliques du graphe. Les cliques maximales s'obtiennent en ne conservant que les cliques qui ne sont incluses dans aucune autre.

Remarquons que, dans le cas d'un appariement, les cliques maximales se calculent sur le graphe d'association, donc $n = |N_1| \times |N_2|$ ce qui devient vite rédhibitoire en temps de calcul. A titre d'exemple, l'appariement de deux graphes d'ordre 6 prend environ 8 secondes avec un processeur Intel(R) Pentium(R)III à 1 266 MHz et 512 MB de RAM, alors que l'appariement de deux graphes d'ordre 8 prend plus d'une minute et demie.

6.3.2.3 Application aux graphes d'attributs, fonction objectif

Considérons maintenant deux graphes d'attributs, $Ga_1 = (N_1, A_1, \sigma_1, \delta_1)$ et $Ga_2 = (N_2, A_2, \sigma_2, \delta_2)$, tels que définis au paragraphe 6.2.1 page 90. La notion de graphe d'association peut se généraliser aux graphes d'attributs en construisant un graphe d'association valué par la similarité inter-éléments. Ainsi chaque élément du graphe d'association généralisé est associé à un poids égal à la similarité entre les deux éléments de Ga_1 et Ga_2 qu'il représente (0 si les arêtes n'existent pas).

Définissons maintenant $Ga = (N, A, \sigma, \delta)$ le graphe d'association généralisé de Ga_1 et Ga_2. En considérant la structure graphique de Ga, il est possible de déterminer une clique maximale C_h et son isomorphisme de sous-graphe associé h. La fonction objectif associée à h est une mesure de similarité globale sur les éléments de la clique. Il existe autant de fonctions objectif que d'algorithmes d'appariement. La plus simple est la moyenne pondérée des similarités des éléments de C_h qui s'exprime de la façon suivante :

$$f_h(Ga_1, Ga_2) = \frac{\gamma}{|N|} \sum_{(a,h(a)) \in C_h} s_N(a, h(a)) + \frac{1-\gamma}{|A|} \sum_{(a,h(a)),(b,h(b)) \in C_h \times C_h} s_A((a, h(a)), (b, h(b)))$$

$$\text{avec } ((a, h(a)), (b, h(b))) \in E \text{ et } \gamma \in [0, 1] \tag{6.6}$$

où le premier terme correspond à la moyenne des similarités des nœuds (ou moyenne des poids des nœuds de Ga) et le second à la moyenne des similarités des arêtes (ou moyenne des poids des arêtes de Ga). Le facteur γ détermine l'importance accordée aux similarités entre arêtes relativement aux similarités entre nœuds. Un appariement possible consiste à retenir l'isomorphisme de sous-graphe qui maximise la fonction objectif. Pour réduire la complexité de la recherche des cliques, il est commode de réduire l'ordre et la taille du graphe d'association généralisé en supprimant les nœuds dont le poids est inférieur à un seuil prédéterminé.

6.3.3 Méthodes locales d'appariement

L'utilisation d'un isomorphisme de sous-graphe avec des graphes d'attributs obtenus à partir d'une fonction objectif pose deux questions. Comment réduire la complexité algorithmique qui rend l'approche globale pratiquement inutilisable ? Comment tolérer une approximation sur l'isomorphisme trouvé pour rendre l'appariement plus robuste au bruit ? Voici deux algorithmes, maintenant classiques, qui permettent, en l'état, ou légèrement modifiés de répondre à ces deux questions.

6.3.3.1 L'algorithme de relaxation floue

Cet algorithme a été défini par RANGANATH ET CHIPMAN ([RC92], [CR92]) afin de rendre l'appariement plus robuste aux bruits. Il se base sur le formalisme du graphe d'association généralisé défini précédemment. Son principe est de recalculer les poids du graphe d'association itérativement pour faire ressortir les nœuds dont les arêtes adjacentes ont un poids élevé avant seuillage. Cette approche, en permettant de ne pas seuiller des nœuds de faible poids dont les arêtes adjacentes représentent une forte

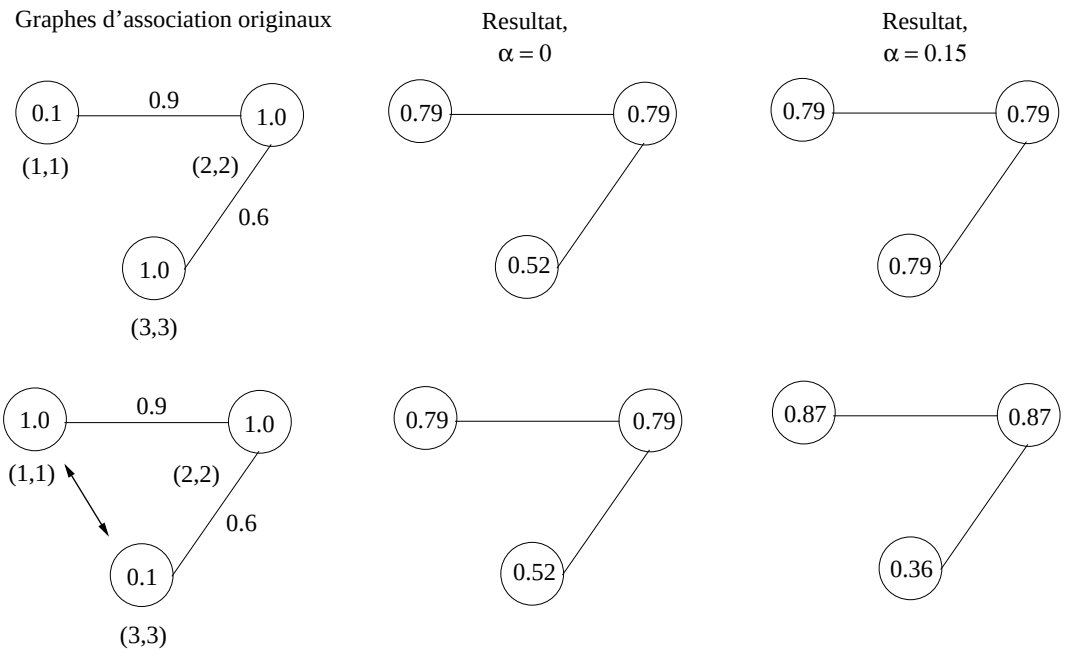


FIGURE 6.6 – Résultats de relaxation avec deux valeurs de α d'après [RC92]. Les graphes d'association de la première et de la deuxième ligne ont des pondérations inversées sur le premier et le troisième nœud.

similarité, rend l'appariement plus robuste aux aspérités du squelette.

Considérons deux graphes attribués non orientés $Ga_1 = (N_1, A_1, \sigma_1, \delta_1)$ et $Ga_2 = (N_2, A_2, \sigma_2, \delta_2)$, leur graphe d'association généralisé peut se représenter par deux matrices S et C représentant respectivement les poids de ses nœuds et de ses arêtes. La matrice S est de dimension $|N_1| \times |N_2|$ et $S_{ab} = s_N(a, b)$ est la similarité entre le a^{e} nœud de Ga_1 et le b^{e} nœud de Ga_2. La matrice C est de dimension $|N_1||N_2| \times |N_1||N_2|$ et $C_{aibj} = s_A(a, b; i, j)$ est la mesure de similarité entre l'arête (a, b) de Ga_1 et l'arête (i, j) de Ga_2, si une des deux arêtes n'existe pas C_{aibj} est mis à zéro.

Le principe de la relaxation est de mettre à jour itérativement les poids de la matrice S en prenant en compte ceux de la matrice C. La méthode la plus répandue pour mettre à jour S est la suivante :

$$S_{ai}^{(r+1)} = \frac{1}{|N_1|} \sum_{b=1}^{|N_1|} \left[\max_{j=1}^{|N_2|} S_{aj}^{(r)} C_{aibj} \right] \tag{6.7}$$

Une fois cette opération réalisée, les poids sont normalisés pour forcer la somme des poids à être constante :

$$S_{ai}^{(r+1)} = \frac{S_{ai}^{(r+1)} \sum_{b=1}^{|N_1|} \sum_{j=1}^{|N_2|} S(b,j)^{(0)}}{\sum_{b=1}^{|N_1|} \sum_{j=1}^{|N_2|} S_{bj}^{(r+1)}} \tag{6.8}$$

Avec cette formule, l'attachement aux données est très faible. Un nœud de poids initialement faible peut obtenir, en fin d'itération, un poids relativement élevé. RANGANATH ET CHIPMAN généralisent cette approche avec une formule permettant de prendre plus en compte les poids initiaux :

$$S_{ai}^{(r+1)} = \alpha S_{ai}^{(0)} + (1 - \alpha) \frac{1}{|N_1|} \sum_{b=1}^{|N_1|} \left[\max_{j=1}^{|N_2|} S_{aj}^{(r)} C_{aibj} \right] \tag{6.9}$$

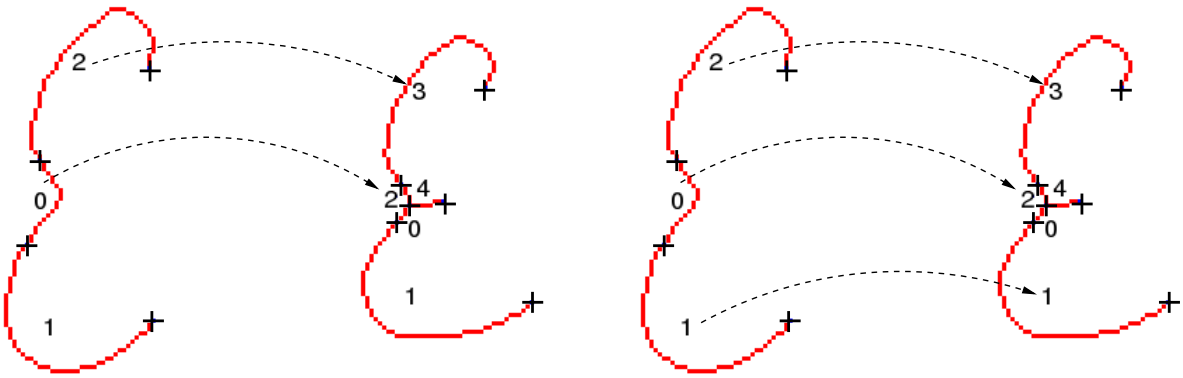
(0,0)=85%	(0,1)=72%	(0,2)=87%	(0,3)=71%	(0,4)=86%
(1,0)=58%	(1,1)=97%	(1,2)=60%	(1,3)=97%	(1,4)=59%
(2,0)=59%	(2,1)=96%	(2,2)=60%	(2,3)=98%	(2,4)=59%

TABLE 6.1 – Matrice S initiale, graphe d'association des caractères de la figure 6.7.

(0,0)=81%	(0,1)=65%	**(0,2)=82%**	(0,3)=65%	(0,4)=80%
(1,0)=62%	**(1,1)=94%**	(1,2)=69%	(1,3)=92%	(1,4)=69%
(2,0)=66%	(2,1)=91%	(2,2)=66%	**(2,3)=94%**	(2,4)=67%

TABLE 6.2 – Matrice S après relaxation floue avec $\alpha = 0,7$, $Tlim = 0,5$, $\delta = 0,01$, graphe d'association des caractères de la figure 6.7. Le choix de l'appariement direct est en gras.

où α est le facteur d'attachement aux données initiales à valeurs réelles dans $[0, 1]$. C'est une valeur dépendante de l'application. Ainsi si $\alpha = 0$, l'équation 6.9 page précédente est équivalente à l'équation 6.7 page précédente et les similarités des arêtes sont prépondérantes dans le résultat d'appariement. La figure 6.6 page précédente, issue de [RC92], montre les résultats de relaxation sur deux graphes d'association généralisés pour lesquels les poids de deux nœuds ont été inversés. Pour $\alpha = 0$ (deuxième colonne) les poids obtenus sont identiques pour les deux graphes d'association, seules les arêtes ont été prises en compte pendant la relaxation. En revanche, pour $\alpha = 0,15$ (troisième colonne), les poids obtenus sont différents, le poids des nœuds a eu une influence pendant la relaxation.



Appariement par clique maximal de similarité maximale

Appariement direct sur les valeurs de S après relaxation

FIGURE 6.7 – Exemple d'appariements par clique maximale et direct.

Le nombre d'itérations nécessaires à la convergence de l'algorithme est dépendant des poids de départ et de la taille du graphe d'association. L'arrêt se fait en comparant tous les poids de $S^{(r+1)}$ et $S^{(r)}$. Si toutes les différences sont inférieures à un seuil fixé δ, le processus est stable et il est arrêté.

RANGANATH ET CHIPMAN proposent ensuite d'utiliser la clique maximale dont la somme des poids des sommets est la plus grande comme isomorphisme de sous-graphe. Il est en effet superflu d'utiliser les similarités entre arêtes comme dans l'équation 6.6 page 96 après la relaxation. Une approximation possible qui s'adapte à notre problématique est d'apparier tous les nœuds du plus petit graphe directement avec les nœuds du plus grand dont la similarité est maximale.

Nous illustrons ces deux approches dans la figure 6.7 page ci-contre. L'appariement de gauche est réalisé avec la clique de poids maximal et celui de droite par mise en correspondance directe des nœuds ayant la similarité maximale sur S. La table 6.1 page précédente donne la matrice S avant relaxation et la table 6.2 page ci-contre après relaxation réalisée avec $\alpha = 0,7$, le seuil sur S après relaxation $Tlim = 0,5$, et le critère de convergence $\delta = 0,01$. Les associations retenues par l'appariement direct sont en gras sur la table 6.2 page précédente. Nous voyons, dans l'exemple donné, que la relaxation a permis de donner plus d'importance à la similarité du couple $(1,1)$ qui avait, dans le graphe d'appariement initial, la même valeur que le couple $(1,3)$ (97% pour les deux dans la table 6.1 page ci-contre contre respectivement 94% et 92% dans la table 6.2 page précédente). L'appariement direct permet de mettre en correspondance le couple $(1,1)$ qui ne pouvait pas être retenu par une approche par clique maximale (puisque les primitives 2 et 1 ne sont pas adjacentes dans le graphe de droite alors que les primitives 0 et 1 le sont dans celui de gauche).

6.3.3.2 L'assignement gradué

L'assignement gradué repose sur une méthode d'optimisation non linéaire proposée par GOLD ET RANGARAJAN pour résoudre le problème d'appariement approximatif de graphes attribués ([GR96]). Les méthodes d'optimisation consistent à exprimer le problème de la recherche d'isomorphisme de graphes par la maximisation d'une fonction objectif (ou minimisation d'une fonction dite alors *d'énergie*). Pour la recherche d'isomorphisme entre deux graphes G_1 et G_2 non attribués non orientés de même ordre n, PELILLO ([Pel99]) donne une formulation continue du problème de recherche de la clique maximale par maximisation de la fonction quadratique suivante :

$$\hat{f}(\mathbf{x}) = \mathbf{x}(\Xi + \frac{1}{2} I_{2n})\mathbf{x} \tag{6.10}$$

où Ξ est la matrice d'adjacence du graphe d'association, I_{2n} est la matrice identité de dimension $n \times n$. Le problème se formule suivant le théorème suivant :

Théorème 5 *Soit $G_1 = (N_1, A_1)$ et $G_2 = (N_2, A_2)$ deux graphes d'ordre n et $\mathbf{x}^\star$ une solution globale de $\max(\hat{f}(\mathbf{x}))$, alors G' et G'' sont isomorphiques si et seulement si $\hat{f}(\mathbf{x}^\star) = 1 - 1/(2n)$. Dans ce cas, $\mathbf{x}^\star$ est un vecteur caractéristique de la clique maximum induite par l'isomorphisme.*

Le vecteur caractéristique d'une clique C est un vecteur $\mathbf{x}^C$ de $\mathbb{R}^{2n}$ tel que :

$$\mathbf{x}_i^C = \begin{cases} 1/|C|, \text{ si } i \in C \\ 0, \text{ sinon.} \end{cases} \tag{6.11}$$

La démonstration du théorème se fait à partir du théorème de Motzkin-Straus, elle est présentée dans l'article [Bom97]. En exprimant la matrice d'adjacence Ξ en fonction des matrices d'adjacences des deux graphes initiaux Ξ_1 et Ξ_2, PELILLO montre que cette optimisation est équivalente à celle proposée par RANGARAJAN ET COLL. ([RM96]) consistant à chercher la matrice M qui minimise la fonction d'énergie suivante :

$$\begin{gathered} E(M) = \sum_{a=1}^{N}\sum_{i=1}^{N}\left(\sum_{b=1}^{N}\Xi_{1ab}M_{bi} - \sum_{j=1}^{N}\Xi_{2ji}M_{aj}\right)^2 \\ \text{avec } \sum_{a=1}^{N} M_{ai} = 1,\ \forall i,\ \sum_{i=1}^{N} M_{ai} = 1,\ \forall a,\ M_{ai} \in [0,1] \end{gathered} \tag{6.12}$$

Pour l'appariement de deux graphes d'attributs, $Ga_1 = (N_1, A_1, \sigma_1, \delta_1)$ et $Ga_2 = (N_2, A_2, \sigma_2, \delta_2)$, GOLD ET RANGARAJAN dérivent de ce critère une fonction d'énergie dépendant des matrices S et C définies au paragraphe précédent :

$$\begin{gathered} E(M) = -\tfrac{1}{2}\sum_{a=1}^{|N_1|}\sum_{i=1}^{|N_2|}\sum_{b=1}^{|N_1|}\sum_{j=1}^{|N_2|} M_{ai}M_{bj}C_{aibj} + \alpha\sum_{a=1}^{|N_1|}\sum_{i=1}^{|N_2|} M_{ai}S_{ai} \\ \text{avec } \sum_{a=1}^{N} M_{ai} = 1,\ \forall i,\ \sum_{i=1}^{N} M_{ai} = 1,\ \forall a,\ M_{ai} \in [0,1] \end{gathered} \tag{6.13}$$

Le premier terme de l'équation 6.13 correspond au binôme de signe négatif du développement de l'équation 6.12 page précédente et exprime la *règle du rectangle* (explicitée dans [GR96]). Le second terme est un terme de contrainte pour tenir compte des similarités entre nœuds et le paramètre α dépend de l'application. La matrice M qui minimise la fonction d'énergie est une matrice de permutation[1]. Elle représente une clique maximale de similarité maximale dans le graphe d'association telle que $M_{ai} = 1$ si le nœud a de Ga_1 est associé avec le nœud i de Ga_2.

L'algorithme d'optimisation par assignement gradué utilisé par GOLD ET RANGARAJAN pour minimiser la fonction d'énergie repose sur le principe suivant. La matrice M est initialisée à une valeur M^0. Son développement en série de Taylor autour de M^0 s'exprime par :

$$E(M) \approx -\frac{1}{2}\sum_{a=1}^{|N_1|}\sum_{i=1}^{|N_2|}\sum_{b=1}^{|N_1|}\sum_{j=1}^{|N_2|} M^0_{ai}M^0_{bj}C_{aibj} - \sum_{a=1}^{|N_1|}\sum_{i=1}^{|N_2|} Q_{ai}(M_{ai} - M^0_{ai}) \tag{6.14}$$

avec

$$Q_{ai} = \frac{\partial E(M)}{\partial M_{ai}} = \sum_{b=1}^{|N_1|}\sum_{j=1}^{|N_2|} M_{bj}C_{aibj} + \alpha S_{ai} \tag{6.15}$$

et la minimisation de $E(M)$ revient à minimiser l'expression $\sum_{a=1}^{|N_1|}\sum_{i=1}^{|N_2|} Q_{ai}M_{ai}$.

Exprimé sous cette forme, le problème est ramené à un problème classique d'assignement. L'assignement gradué présente l'avantage de réduire la complexité. Les éléments de la matrice M sont exprimés à chaque itération en fonction de ceux de la matrice Q en introduisant un paramètre de contrôle β :

$$M_{ai} = \exp(\beta Q_{ai}) \tag{6.16}$$

Ensuite, ils sont normalisés successivement par la somme des éléments des lignes puis des colonnes de M :

$$M_{ai} = \frac{M_{ai}}{\sum_{i=1}^{|N_2|} M_{ai}} \qquad \text{puis} \qquad M_{ai} = \frac{M_{ai}}{\sum_{a=1}^{|N_1|} M_{ai}} \tag{6.17}$$

Q est alors recalculée et le processus est répété avec la même valeur de β jusqu'à ce que la matrice M soit stabilisée :

$$\sum_{a=1}^{|N_1|}\sum_{i=1}^{|N_2|} |M^0_{ai} - M_{ai}| < \delta_0 \text{ ou bien le nombre d'itérations } I \geq I_0 \tag{6.18}$$

Enfin β est mis à jour par un facteur multiplicatif $\beta = \beta \times \beta_r$ puis l'ensemble du calcul est répété. Le critère d'arrêt est assuré par un seuil β_f sur β. Pour assurer la convergence, il faut ajouter un bouclage itératif sur l'équation 6.17 jusqu'à convergence de la matrice M :

$$\sum_{a=1}^{|N_1|}\sum_{i=1}^{|N_2|} |M^{(k)}_{ai} - M^{(k+1)}_{ai}| < \delta_1 \text{ ou le nombre d'itérations } I \geq I_1 \tag{6.19}$$

[1]Les éléments d'une matrice de permutation valent 0 ou 1 et la somme des éléments par colonne et par ligne vaut 1

(0,0)=1	(0,1)=0	(0,2)=0	(0,3)=0	(0,4)=0
(1,0)=0	(1,1)=1	(1,2)=0	(1,3)=0	(1,4)=0
(2,0)=0	(2,1)=0	(2,2)=1	(2,3)=0	(2,4)=0

TABLE 6.3 – Matrice M_0 au début de l'algorithme 3 page suivante pour l'exemple de la figure 6.8.

(0,0)=60%	(0,1)=0%	**(0,2)=99%**	(0,3)=0%	(0,4)=0%	(0,5)=0%
(1,0)=6%	(1,1)=54%	(1,2)=0%	(1,3)=24%	**(1,4)=86%**	(1,5)=0%
(2,0)=33%	(2,1)=45%	(2,2)=0%	**(2,3)=75%**	(2,4)=13%	(2,5)=0%
(3,0)=0%	(3,1)=0%	(3,2)=0%	(3,3)=0%	(3,4)=0%	(3,5)=100%

TABLE 6.4 – Matrice $\hat{M}$ à la fin de l'algorithme 3 page suivante pour l'exemple de la figure 6.8. L'appariement donne les associations mises en gras.

La convergence de cette étape est basée sur une extrapolation du résultat donné par SINKHORN ([Sin64]) sur la convergence d'une matrice carrée positive vers une matrice doublement stochastique[2]. Pour pallier le fait que M n'est pas carrée, GOLD ET RANGARAJAN proposent de lui ajouter, pour cette étape, une ligne et une colonne afin de permettre les non appariements ($\hat{M}$ est la matrice M avec une ligne et une colonne de plus).

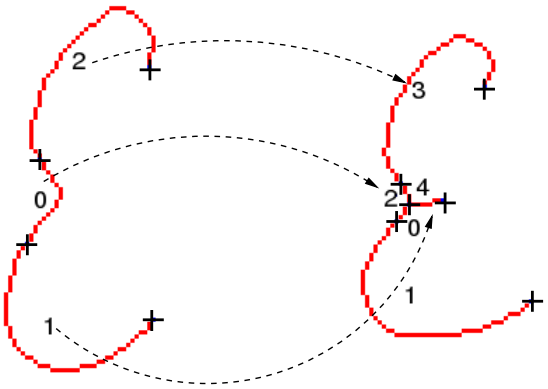


FIGURE 6.8 – Exemple d'appariement par assignement gradué.

L'utilisation du résultat de SINKHORN assure une relativement bonne convergence vers une matrice doublement stochastique, en revanche la matrice M en fin de processus n'est pas systématiquement une matrice de permutation. Pour s'en assurer, GOLD ET RANGARAJAN ajoutent une heuristique finale qui met à 1 les valeurs maximales de chaque colonne et à 0 toutes les autres. Cette heuristique est fondée puisqu'une matrice doublement stochastique ne peut avoir, par définition, les maxima de deux colonnes dans la même ligne. Mais comme les deux convergences précédentes ne sont qu'approximatives, nous avons choisi une heuristique plus stricte qui met, une à une, les valeurs maximales de M à 1 en mettant simultanément à 0 toutes les autres valeurs de la même ligne et de la même colonne.

L'algorithme 3 page suivante récapitule le processus global de l'assignement gradué. Sa complexité est en $\mathcal{O}(|A_1||A_2|)$ ce qui le rend très intéressant malgré les imprécisions de convergence. Nous illustrons

[2]Une matrice doublement stochastique est une matrice définie positive dont la somme des éléments en colonne et en ligne vaut 1

Algorithme 3: Algorithme d'assignement gradué de GOLD ET RANGARAJAN

Données : M^0 la matrice de permutation initiale
S et C les matrices de poids du graphe d'association
Les variables β_0, β_f, β_r, δ_0, I_0, δ_1, I_1

Résultat : M la matrice de permutation finale

début
- $M \leftarrow M^0$;
- $\beta \leftarrow \beta_0$;
- **répéter**
  - **répéter**
    - Calcul de Q suivant l'équation 6.15 page 100;
    - Calcul de M suivant l'équation 6.16 page 100;
    - $\hat{M} \leftarrow M$ + une ligne de 0 + une colonne de 0;
    - **répéter**
      - Normalisation de $\hat{M}$ suivant l'équation 6.17 page 100;
    - **jusqu'à** *($\hat{M}$ satisfait l'équation 6.19 page 100)*;
    - $M \leftarrow \hat{M}$;
  - **jusqu'à** *(M satisfait l'équation 6.18 page 100)*;
  - $\beta \leftarrow \beta \times \beta_r$;
- **jusqu'à** *($\beta > \beta_f$)*;

fin

l'algorithme sur un exemple à la figure 6.8 page précédente, la matrice $\hat{M}$ à la fin de l'appariement est donnée à la table 6.4 page précédente, l'appariement retenu après l'heuristique finale est noté en gras. Le calcul a été effectué avec les paramètres suivants : $\alpha = 0,5$, $\beta_0 = 0,5$, $\beta_f = 10$, $\beta_r = 1,075$, $\delta_0 = 0,5$, $\delta_1 = 0,05$, $I_0 = 10$, $I_1 = 30$. Le résultat est conforme à celui obtenu avec l'algorithme de relaxation floue et recherche de clique maximale, avec la mise en correspondance $(1, 4)$ en plus qui avait été seuillée avec ce dernier (figure 6.7 page 98 (a)). La mise en correspondance $(1, 1)$ n'a pas été retenue puisqu'elle ne fait pas partie de la clique maximale. L'initialisation M_0 joue un rôle important, nous avons choisi celle proposée par GOLD ET RANGARAJAN présentée à la table 6.3 page précédente.

L'algorithme par relaxation floue comme celui par assignement gradué est basé sur la recherche d'une clique maximale dans le graphe d'association généralisé. Utilisés avec une recherche directe des appariements dans les matrices résultats, ils permettent tous les deux de dépasser la simple clique maximale et d'apparier tous les nœuds du graphe de plus petite cardinalité. Malgré tout le résultat contient souvent au moins partiellement la clique maximale, ce qui est un compromis intéressant d'autant plus que la complexité est raisonnable. Ils sont, en résumé, relativement bien adaptés à notre problématique et leurs performances relatives seront analysées dans le chapitre 7 page 107.

6.3.4 Appariements des arêtes

L'appariement, qu'il soit réalisé avec l'algorithme de relaxation floue ou d'assignement gradué, fournit une mise en correspondance des nœuds. Il est intéressant pour la suite d'obtenir également une mise en correspondance des arêtes. Compte tenu de la nature des graphes traités qui sont multiples et non orientés dans le cas général, l'heuristique utilisée est simple. Un double balayage de la liste des couples de nœuds appariés est réalisé (sans exclusion mutuelle pour pouvoir apparier les boucles) et toutes les arêtes existantes entre les deux couples de nœuds considérés sont mises en correspondance suivant le maximum de similarité. Une arête ne peut être appariée qu'une seule fois.

6.4 Principe de reconnaissance

L'apprentissage consite à apparier tous les graphes d'attributs d'une même classe avec un graphe modèle. Ainsi chaque classe est maintenant représentée par son graphe modèle dont les éléments sont associés à une série d'observations, attributs des graphes de la classe.

Il est possible d'associer à chaque élément du graphe modèle d'une classe une loi de distribution de probabilités multivariée calculée à partir des observations. Chaque classe est alors représentée par un graphe dont le formalisme est analogue à celui des graphes aléatoires définis par WONG ET YOU ([WY85], [Won87]). L'adoption de ce formalisme nous permet d'utiliser certains résultats de ces articles pour aborder la reconnaissance.

6.4.1 Les graphes aléatoires

Dans l'article [WY85], WONG ET YOU explicitent les graphes aléatoires comme un graphe dont chaque élément est une variable aléatoire :

Définition 30 Graphe aléatoire *Un graphe aléatoire est une paire $R = (W, B)$ telle que :*

1. *W, représentant l'ensemble des* nœuds aléatoires*, est un n-uplet $(\alpha_1, \alpha_2, \ldots, \alpha_n)$ où chaque α_i, appelé nœud aléatoire, est une variable aléatoire ;*
2. *B, représentant l'ensemble des* arêtes aléatoires*, est un m-uplet $(\beta_1, \beta_2, \ldots, \beta_m)$ où chaque β_i, appelé arête aléatoire, est une variable aléatoire ;*
3. *Pour chaque graphe $G = (N, A)$ réalisation possible de R associé à un isomorphisme de sous-graphe, $\phi : G \rightarrow R$, il existe une probabilité $p(G, \phi) = Pr(R = \phi(G), \phi \in \Phi)$. L'ensemble Φ est la famille des isomorphismes de sous-graphe tels que :*

$$\begin{aligned} p(G, \phi) \geq 0,\ \forall G \in \Gamma \\ \sum_{\Gamma} p(G, \phi) = 1 \end{aligned} \tag{6.20}$$

Notons que WONG ET YOU utilisent des variables discrètes dont les probabilités sont calculées de manière fréquentielle. L'extension aux graphes d'attributs, à valeurs continues dans notre cas, se fait naturellement à partir des définitions données au paragraphe 6.2.1 page 90. Si les $z_i \in Z$ sont considérés comme des variables aléatoires à valeurs dans $[0, 1]$ avec une loi de probabilité associée f_{z_i} et les $f_i \in F$ sont des variables aléatoires à valeurs dans $[0, 1]$ avec une loi probabilité associée f_{f_i}, alors le graphe d'attributs devient un graphe aléatoire.

Ainsi définis, les graphes d'attributs modèles peuvent être considérés comme des graphes aléatoires. Nous associons à chaque attribut une loi de probabilité estimée à partir des observations de la base d'apprentissage. Les lois sont supposées gaussiennes diagonales et les espérances comme les variances sont estimées statistiquement. Ce choix repose sur deux hypothèses : la normalité et l'indépendance des attributs. La première hypothèse est justifiée par la nature continue des données et l'hétérogénéité des bases. Cependant les tests de normalités d'Anderson-Darling ([Ste74]) par exemple, montrent qu'elle est rarement respectée (un exemple est donné dans le chapitre suivant). Il n'y a là rien d'étonnant vu les approximations effectuées par les algorithmes d'appariement. De plus l'hypothèse d'indépendance n'est pas toujours vérifiée (la rectitude et la longueur d'un segment primitif par exemple ne sont pas décorrélées). Malgré tout, en l'état, l'hypothèse d'indépendance donne empiriquement de meilleurs résultats que l'utilisation d'une loi gaussienne quelconque.

Nous supposons donc que les nœuds α_i des graphes d'attributs modèles sont tels que $\alpha_i \rightsquigarrow \mathcal{N}(\boldsymbol{\mu}_{\alpha_i}, \Sigma_{\alpha_i})$ et les arêtes $\beta_i = (\alpha_l, \alpha_m)$ sont telles que $\beta_i \rightsquigarrow \mathcal{N}(\boldsymbol{\mu}_{\beta_i}, \Sigma_{\beta_i})$ avec Σ_{α_i} et Σ_{β_i} diagonales.

Soit $\{R_k = (W_k, B_k) | k = 1 \dots c\}$ les graphes aléatoires modèles des classes C_k de la base d'apprentissage et $Ga = (N, A, \sigma, \delta)$ un graphe d'attributs inconnu à reconnaître. La reconnaissance se fait en deux étapes. La première consiste à trouver un isomorphisme approximatif de sous-graphe entre le graphe inconnu et chacun des graphes aléatoires. La seconde étape est celle de la décision.

6.4.2 Appariement

Avec le même formalisme que pour la classification, la recherche de l'isomorphisme approxiamatif de sous-graphe passe par la construction d'un graphe d'association généralisé entre le graphe d'attributs et chacun des graphes aléatoires. Les poids associés aux éléments du graphe d'association généralisé peuvent être calculés de plusieurs manières.

L'approche la plus simple est d'apparier G_a avec les graphes d'attributs moyens issus des R_k qui possèdent la structure graphique des modèles et dont les attributs sont les moyennes des observations. Ignorer les variances des distributions présente l'avantage de minimiser l'influence des approximations d'appariement.

Une approche plus complète serait d'utiliser la vraisemblance exprimée par la densité de probabilité qu'un nœud a (représenté par son vecteur d'attribut $\boldsymbol{\alpha}_a$) de G_a soit une réalisation d'un nœud α_i de R_k ($f(\boldsymbol{\alpha}_a | \boldsymbol{\mu}_{\alpha_i}, \Sigma_{\alpha_i})$) mais elle n'est pas homogène à une similarité (elle n'est pas à valeurs dans $[0, 1]$). Une discrétisation du problème en utilisant la probabilité *a posteriori* de l'indice i connaissant a, estimée par la règle de Bayes, n'avancerait à rien puisque nous ne pouvons pas évaluer la densité $f(a)$.

Nous proposons donc d'utiliser la probabilité de dispersion de α_i par rapport à a basée sur la distance de Mahalanobis. Nous rappelons sa définition :

Définition 31 Probabilité de dispersion *Soit X une variable aléatoire sur $\mathbb{R}^2$ de loi de distribution normale $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$. Soit $\boldsymbol{\alpha}$ un point de $\mathbb{R}^2$. La probabilité de dispersion de X par rapport à $\boldsymbol{\alpha}$ est la probabilité qu'une observation $\boldsymbol{y}$ de X soit située en dehors de l'ellipse de dispersion de niveau p_Δ (proposition 2 page 39) où Δ est la distance de Mahalanobis entre $\boldsymbol{\mu}$ et $\boldsymbol{\alpha}$ ($\Delta = d_M(\boldsymbol{\alpha}, \boldsymbol{\mu})$). Elle est donnée par :*

$$P(d_M(\boldsymbol{y}, \boldsymbol{\mu}) \geq \Delta) = 1 - p_\Delta = e^{\frac{-\Delta}{2}} \tag{6.21}$$

La demonstration de l'équation 6.21 en dimension 2 est donnée dans la thèse de DANG ([Dan98]). En dimension 1 le résultat est classique et correspond aux tables de normalités. Nos attributs étant de dimension 1 ou 2, nous pouvons utiliser ce résultat et écrire :

$$P(d_M(\boldsymbol{y}, \boldsymbol{\mu}_{\alpha_i}) \geq P(d_M(\boldsymbol{\alpha}_a, \boldsymbol{\mu}_{\alpha_i}))) = \exp\left(-\frac{1}{2}(\boldsymbol{\alpha}_a - \boldsymbol{\mu}_{\alpha_i})^\perp {\Sigma_{\alpha_i}}^{-1} (\boldsymbol{\alpha}_a - \boldsymbol{\mu}_{\alpha_i})\right) \tag{6.22}$$

Notons que, dans notre cas, les Σ_i étant diagonales, nous pouvons toujours calculer cette probabilité axe par axe et donc garder ce résultat vrai quelle que soit la dimension.

En pondérant cette probabilité par le rapport π_{α_i} entre le nombre d'observations associées au nœud α_i et le nombre total d'observations associées à R_k (égal au nombre de caractères de la classe dans la base), nous obtenons une mesure homogène à la probabilité discrète utilisée par WONG ET YOU que a soit une réalisation de α_i ($Pr(\alpha_i = \boldsymbol{\alpha}_a)$). Pour rester dans le formalisme WONG ET YOU, nous avons conservé cette notation :

$$Pr(\alpha_i = \boldsymbol{\alpha}_a) = \pi_{\alpha_i} \times \exp\left(-\frac{1}{2}(\boldsymbol{\alpha}_a - \boldsymbol{\mu}_{\alpha_i})^\perp {\Sigma_{\alpha_i}}^{-1} (\boldsymbol{\alpha}_a - \boldsymbol{\mu}_{\alpha_i})\right) \tag{6.23}$$

De façon similaire, si $\beta_i = (\alpha_l, \alpha_m)$ est une arête de R_k et (a, b) une arête de G_a (représentée par son vecteur d'attribut $\boldsymbol{\beta}_{ab}$) , la probabilité $Pr(\beta_i = (\boldsymbol{\beta}_{ab}))$ s'écrit :

$$Pr(\beta_i = \boldsymbol{\beta}_{ab}) = \pi_{\beta_i} \times \exp\left(-\frac{1}{2}(\boldsymbol{\beta}_{ab} - \boldsymbol{\mu}_{\beta_i})^{\perp}{\Sigma_{\beta_i}}^{-1}(\boldsymbol{\beta}_{ab} - \boldsymbol{\mu}_{\beta_i})\right) \tag{6.24}$$

avec π_{β_i} représentant le nombre d'observations associées à l'arête β_i sur le nombre total d'observations associées à R_k (égal au nombre de caractères de la classe dans la base). Ensuite, en remplaçant les similarités par les probabilités *a posteriori* de dispersion dans les matrices S et C, les algorithmes d'appariement par relaxation floue ou par assignement gradué donnent un isomorphisme approximatif de sous-graphe entre Ga et chacun des R_k. La première étape du processus de reconnaissance est terminée.

6.4.3 Décision

La phase de décision utilise un résultat donné par WONG ET YOU sur les graphes aléatoires. Il repose sur trois hypothèses :

1. les nœuds aléatoires d'un graphe aléatoire sont mutuellement indépendants ;
2. les arêtes aléatoires d'un graphe aléatoire sont indépendantes de tous les nœuds du graphe exceptés les deux qui forment ses extrémités ;
3. les distributions conditionnelles des arêtes aléatoires relatives aux nœuds aléatoires d'un graphe aléatoire sont mutuellement indépendantes.

Sous ces hypothèses, WONG ET YOU expriment la probabilité qu'un graphe $G = (N, A)$ soit la réalisation d'un graphe aléatoire $R = (W, B)$ par l'isomorphisme de sous-graphe ϕ de la façon suivante :

$$p(G, \phi) = \prod_{\alpha_i \in W} Pr\{\alpha_i = \phi^{-1}(\alpha_i)\} \times \prod_{\beta_i = (\alpha_l, \alpha_m) \in B} Pr\{\beta = \phi^{-1}(\beta)) | \alpha_l = \phi^{-1}(\alpha_l),\ \alpha_m = \phi^{-1}(\alpha_m)\} \tag{6.25}$$

Pour un graphe aléatoire à variables continues, cette probabilité n'a pas de sens. Par contre la grandeur équivalente exprimée en fonction des probabilités de dispersion est une mesure de comparaison valable entre un graphe d'attributs et un graphe aléatoire. Nous continuerons de l'appeler probabilité de réalisation par analogie avec le formalisme de WONG ET YOU mais elle n'exprime en réalité qu'une ressemblance entre Ga et les réalisations de R. Ainsi, la probabilité que $Ga = (N, A, \sigma, \delta)$ soit une réalisation de $R_k = (W_k, B_k)$ par ϕ s'exprime :

$$p(Ga, \phi) = \prod_{\alpha_i \in W} Pr\{\alpha_i = \phi^{-1}(\alpha_i)\} \times \prod_{\beta_i = (\alpha_l, \alpha_m) \in B} Pr\{\beta = \phi^{-1}(\beta)) | \alpha_l = \phi^{-1}(\alpha_l),\ \alpha_m = \phi^{-1}(\alpha_m)\} \tag{6.26}$$

Le problème de cette formulation est qu'elle ne tient pas compte des nœuds et arêtes de Ga qui n'ont pas d'image par ϕ^{-1}. WONG ET YOU considèrent, à juste titre, ces cas comme impossibles dans leur formalisme et attribuent alors une probabilité nulle à la réalisation Ga. Dans notre cas, le choix des graphes modèles nous oblige à en tenir compte. Il nous faut donc pondérer la probabilité par le nombre d'éléments de Ga non appariés. Nous avons multiplié empiriquement $p(Ga, \phi)$ par un facteur égal à $0,1^{Na}$ où Na est le nombre d'éléments non appariés.

Une fois $p(Ga, \phi)$ calculée pour chaque R_k, la décision est prise par la règle du maximum de vraisemblance :

$$Ga \in C_i \Leftrightarrow i = \arg\max_k(\pi_k \times p(Ga, \phi)) \tag{6.27}$$

où $\pi_k = \frac{m_k}{m}$ est le rapport du nombre de caractères de la classe k sur le nombre total d'éléments dans la base.

Voyons maintenant quelles peuvent être les performances en reconnaissance d'un tel système. C'est l'objet du prochain chapitre.

CHAPITRE 7

RÉSULTATS ET DISCUSSION

Sommaire

Nous présentons ici les résultats de classification obtenus sur les bases de caractères avec les deux types de graphes (primaires et d'arêtes) et les deux algorithmes d'appariement (relaxation floue et assignement gradué). Nous comparerons les différentes combinaisons et essaierons de mettre en évidence les apports de ces méthodes par rapport aux méthodes statistiques dans le cadre d'une mise en coopération basée sur le rejet en ambiguïté de DUBUISSON présenté dans le paragraph 3.3 page 47 ([Dub01]).

Concernant les paramètres des algorithmes d'appariement, seules les valeurs suivantes ont été testées :

- pour l'algorithme de GOLD ET COLL. par assignement gradué,
  - le facteur de pondération du poids des nœuds dans la fonction d'énergie, $\alpha = 0,5$,
  - la valeur initiale du paramètre de contrôle, $\beta_0 = 0,5$,
  - la valeur maximale du paramètre de contrôle, $\beta_f = 10$,
  - le taux de croissance du paramètre de contrôle, $\beta_r = 1,075$,
  - le critère de convergence pour chaque valeur de β, $\delta_1 = 0,05$,
  - le maximum d'itérations autorisées pour chaque valeur de β, $I_0 = 10$,
  - le critère de stabilisation sur la normalisation de M_{ai}, $\delta_0 = 0,5$,

  - le maximum d'itérations autorisées pour la stabilisation de M_{ai}, $I_1 = 30$,
- et pour l'algorithme de RANGARAJAN ET COLL. par relaxation floue,
  - le facteur d'attachement aux données initiales, $\alpha = 0,7$,
  - le seuil des poids du graphe d'association après relaxation, $Tlim = 0,5$,
  - le critère de convergence pour la relaxation, $\delta = 0,01$.

Les paramètres de l'algorithme de GOLD ET COLL. sont simplement ceux utilisés par les auteurs dans leur évaluation ([GR96]). Les paramètres de l'algorithme de RANGARAJAN ET COLL. ont été choisis après une étude comparative des résultats de classification sur la base GrAnc.

7.1 Classifications simples

Nous commençons par donner les résultats de classification sur les différentes bases de caractères pour chacun des algorithmes et les deux types de graphe. Pour la base GrCor nous présentons également les résultats obtenus avec un appariement réalisé par similarité des éléments du graphe d'attribut avec la moyenne des éléments aléatoires et avec un appariement réalisé à partir des probabilités de dispersion.

7.1.1 Graphes primaires

7.1.1.1 Base GrCor

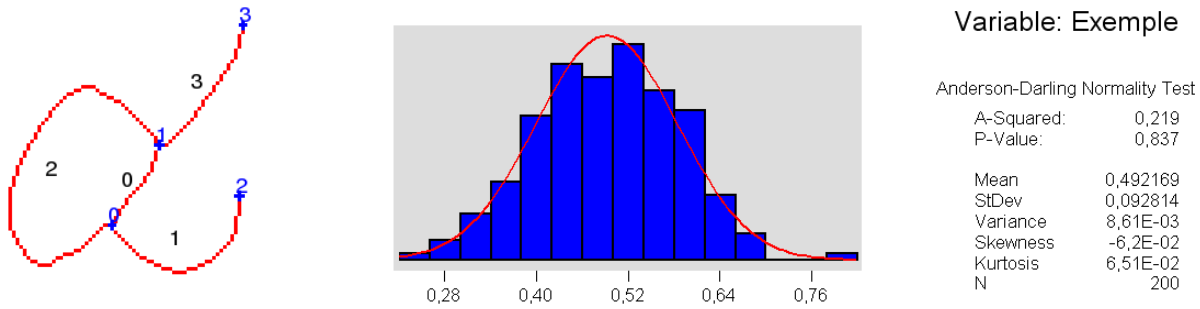
Comme pour les tests réalisés avec les techniques statistiques, les résultats de classification ont été obtenus par validation croisée. Quatre mille caractères ont été tirés au hasard dans la base pour former les bases de tests. Ces 4 000 caractères de test ont été divisés en quatre groupes de 1 000 individus. Pour chaque groupe, un taux d'erreur de classification a été calculé par rapport à une base d'apprentissage composée des 3 000 individus des trois autres groupes de test plus le reste de la base. La table 7.1 donne la moyenne des erreurs sur les quatre groupes ainsi que l'écart type.

Algorithmes d'Appariement	GOLD ET COLL.	RANGARAJAN ET COLL.
Appariement par moyenne		
Erreur Moyenne	47,4%	52,9%
Ecart Type	0,9%	1,6%
Appariement par probabilité de dispersion		
Erreur Moyenne	65,3%	64,7%
Ecart Type	1,6%	0,8%

TABLE 7.1 – Classification par graphes aléatoires primaires avec la base GrCor

Constatons tout d'abord que l'algorithme de GOLD ET COLL. donne des résultats sensiblement meilleurs dans les deux cas. Ensuite, à l'évidence l'utilisation de la probabilité de dispersion pour l'appariement donne des résultats nettement moins bons qu'avec une simple similarité par rapport aux moyennes. Cela est dû à l'hypothèse de normalité sur les éléments. Pour illustrer ce propos prenons l'exemple du 'α' minuscule. Le graphe modèle possède la structure présentée sur la figure 7.1 page suivante (a). Les nœuds sont les points singuliers numérotés de 0 à 3 en bleu et les arêtes sont les primitives numérotées de 0 à 3 en noir.

La valeur de p dans le test de normalité d'Anderson-Darling ([Ste74]) donne une estimation de la confiance qu'une série d'observations suive une distribution gaussienne. Sur la figure 7.1 page ci-contre (b) nous présentons un exemple sur une série de 200 observations générées alétoirement suivant une loi gaussienne d'espérance $0,5$ et d'écart type $0,1$. Nous calculons une valeur de $p = 0,837$ ce qui donne



(a) (b)

FIGURE 7.1 – (a) structure du modèle de la classe 'α', (b) exemple de test de normalité pour une série de 200 points générés aléatoirement suivant $\mathcal{N}(\mu = 0,5; \sigma = 0,1)$

une bonne confiance pour valider l'hypothèse gaussienne. Il est d'usage de réfuter l'hypothèse quand $p < 0,05$. Nous présentons les mêmes tests réalisés sur tous les attributs du graphe aléatoire primaire de la classe 'α' dans la table 7.2 page suivante. L'appariement pour ce graphe aléatoire, est réalisé avec l'algorithme de GOLD ET COLL. par rapport aux moyennes. Le résultat est sans équivoque puisque la valeur p est nulle pour tous les attributs. Autrement dit l'hypothèse gaussienne n'est jamais vérifiée. Remarquons toutefois que pour le deuxième attribut des nœuds qui est un angle, le test n'utilise pas la soustraction périodique ce qui peut fausser le calcul mais cela ne change pas la conclusion. Nous arrivons là aux limites du modèle proposé. Dans la suite des résultats présentés, nous ne considérerons plus que l'appariement par rapport aux moyennes des éléments.

7.1.1.2 Base MNIST

La table 7.3 page 111 donne les résultats obtenus par validation croisée sur quatre groupes de 1 000 individus.

Sur cette base les algorithmes semblent équivalents avec un léger avantage pour celui de GOLD ET COLL.. La table 7.4 page 111 présente la matrice de confusion avec l'appariement de GOLD ET COLL., avec les pourcentages de bonnes reconnaissances. Ces pourcentages sont la moyenne sur les quatre groupes et le nombre total cumulé d'éléments inconnus par classe est donné dans la dernière colonne. La dernière ligne donne la pertinence de la classe suivant la formule :

$$P = \begin{cases} \frac{\alpha}{\alpha+\gamma} \text{ si } \alpha + \gamma > 0, \\ 1 \text{ sinon} \end{cases} \tag{7.1}$$

avec α le pourcentage de caractères bien détectés sur la colonne et γ la somme des pourcentages des autres éléments de la colonne.

Avant de commenter la matrice de résultat, nous donnons les structures des chiffres de la base pour mieux visualiser. Les squelettes des graphes modèles sont donnés dans la table 7.5 page 111, les squelettes sont en gris et les points singuliers sont symbolisés par une croix noire.

Revenons maintenant sur la matrice de confusion. Le '0' et le '1' sont les chiffres les mieux reconnus. En regardant la structure du modèle, il est assez facile d'interpréter ce résultat en constatant que leurs structures sont très simples donc facilement appariables et que les caractéristiques des arêtes sont très discriminantes. Par contre le '0' est très peu pertinent, les chiffres '1', '3', '6', '8' et '9' sont assez facilement confondus avec lui. Pour le '1' il s'agit des écritures courbes, quand le segment est trop courbé il finit par être confondu avec '0' qui a alors la même structure. Pour les quatre autres caractères cela vient

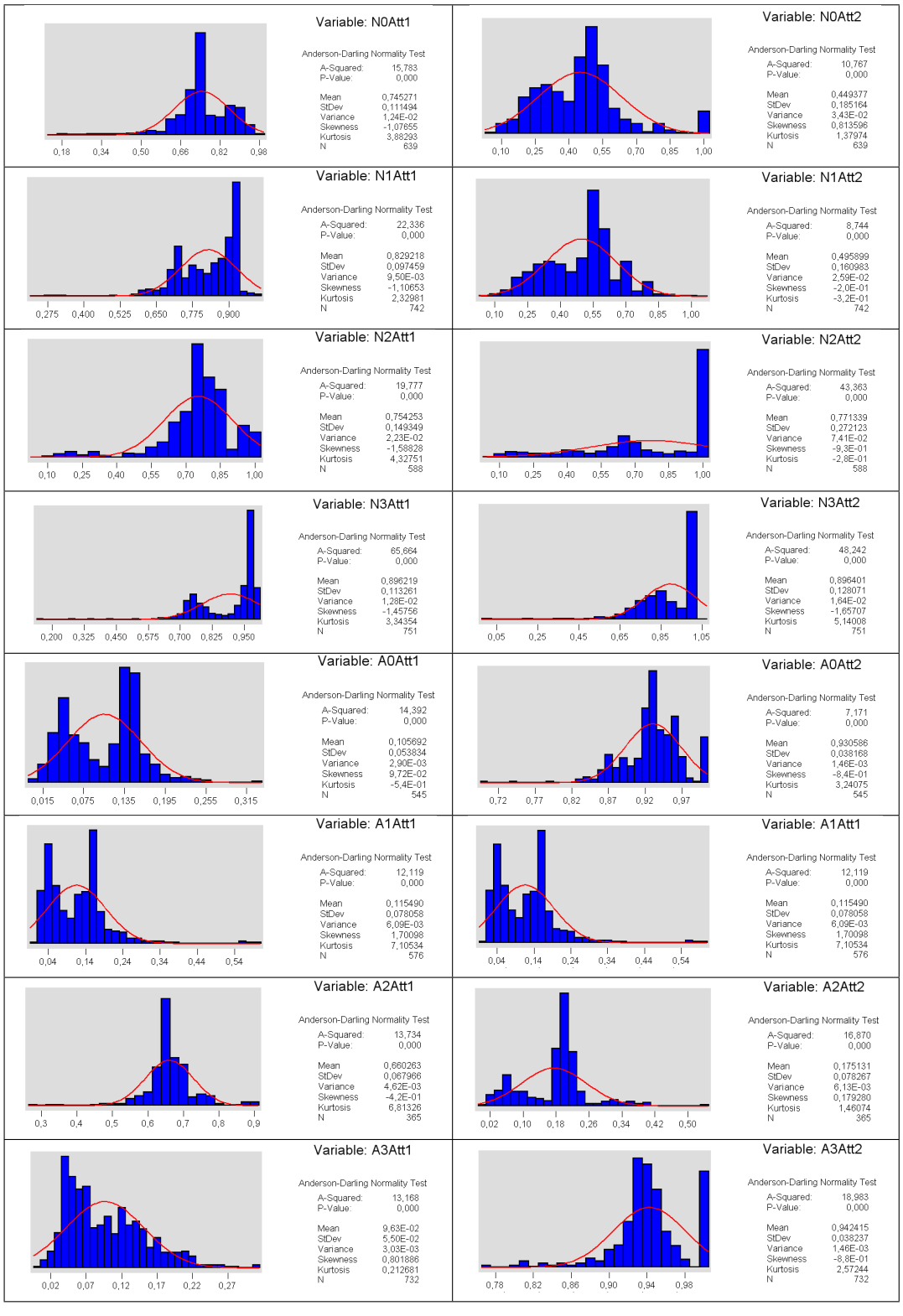


TABLE 7.2 – Tests de normalité réalisés sur les attributs du graphe aléatoire primaire de la classe ’α’. Le nom de la variable de chaque test donne le type d’élément (N pour les nœuds et A pour les arêtes), son numéro dans la structure (comme sur la figure 7.1 page précédente (a)) et le numéro d’attribut (Att1 $= \rho$, Att2 $= \phi$ pour les nœuds et Att1 $= lr_{ab}$, Att $= St_{ab}$ pour les arêtes)

certainement de la boucle (souvent présente également sur les ’3’). Un moyen de résoudre ce problème serait d’augmenter le facteur de pondération de $0,1^{Na}$ sur la probabilité de réalisation (Na est le nombre

Algorithmes d'Appariement	GOLD ET COLL.	RANGARAJAN ET COLL.
Erreur Moyenne	51,0%	57,8%
Ecart Type	1,3%	1,0%

TABLE 7.3 – Classification par graphes aléatoires primaires avec la base MNIST

Classes	0	1	2	3	4	5	6	7	8	9	Nombre d'éléments
0	**89**	1	1	2	1	0	2	0	0	5	392
1	11	**86**	0	0	1	0	1	0	0	0	444
2	4	0	**39**	13	6	0	12	3	0	23	422
3	19	0	8	**50**	2	5	4	8	0	5	387
4	2	1	19	7	**36**	0	14	0	0	20	401
5	1	0	3	25	2	**32**	32	1	0	3	367
6	32	1	5	6	10	1	**40**	1	0	5	380
7	1	0	2	7	7	0	4	**56**	0	23	414
8	13	1	10	15	9	0	17	3	**0**	33	385
9	12	2	4	3	6	0	6	11	0	**55**	408
Pertinence	0,48	0,93	0,43	0,4	0,46	0,83	0,3	0,67	1	0,32	

TABLE 7.4 – Matrice de confusion (en pourcentage de bonnes reconnaisssances) pour la classification par graphes aléatoires primaires avec la base MNIST et l'algorithme de GOLD ET COLL.. Les colonnes correspondent aux valeurs reconnues, les lignes aux valeurs vraies.

d'éléments non appariés, voir paragraphe 6.4.3 page 105).

Le deuxième constat frappant en regardant la matrice de confusion est que le '8' n'est jamais reconnu. Il y a deux raisons à cela. Tout d'abord la structure primaire est très sensible à la présence de boucles qui change la structure même du graphe. Or les structures des '8' peuvent, comme pour le modèle, ne pas avoir de boucle ou bien en avoir une ou encore deux suivant les écritures. La seconde raison est le bruit de structure qui est souvent présent sur les '8' justement à cause des boucles qui, si elles sont trop fermées, provoquent des changements structurels importants.

Enfin pour les autres caractères, là encore ce sont essentiellement les bouclages qui expliquent les confusions : le '5' avec le '6', le '4' avec le '9'...

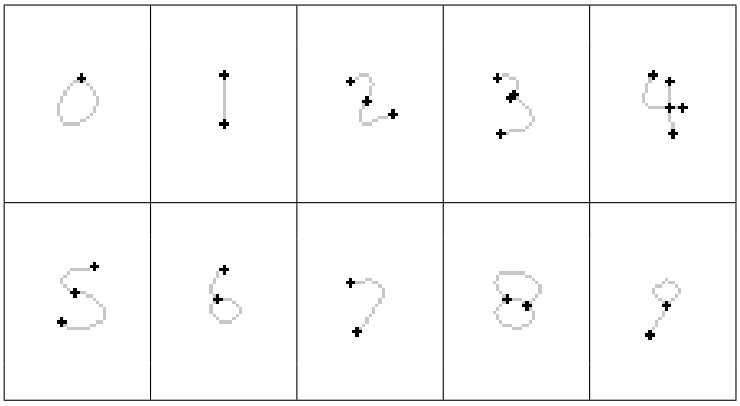


TABLE 7.5 – Structure des modèles de la base MNIST.

7.1.1.3 Base GrAnc

La table 7.6 présente les résultats de classification par graphes aléatoires primaires obtenus par validation croisée sur quatre groupes de 200 individus.

Algorithmes d'Appariement	GOLD ET COLL.	RANGARAJAN ET COLL.
Erreur Moyenne	51,7%	57,5%
Ecart Type	1,3%	4,0%

TABLE 7.6 – Classification par graphes aléatoires primaires avec la base GrAnc

Un fois de plus les résultats sont sensiblement meilleurs avec l'alorithme de GOLD ET COLL. et ce en moyenne comme en écart type. Notons tout de même que les scores sont comparables à ceux obtenus avec la base GrCor ce qui tend à montrer que la structure reste discriminante malgré le bruit inhérant aux caractères anciens. D'une manière générale, les résultats obtenus avec les trois bases sont comparables. La complexité que nous avions mise en avant avec les résultats de reconnaissance statistique ne semble pas avoir d'influence ici.

7.1.2 Graphes d'arêtes

7.1.2.1 Base GrCor

La table 7.7 présente les résultats obtenus avec les graphes d'arêtes dans les mêmes conditions qu'avec les graphes primaires.

Algorithmes d'Appariement	GOLD ET COLL.	RANGARAJAN ET COLL.
Appariement par moyenne		
Erreur Moyenne	63,8%	63,7%
Ecart Type	1,9%	2,6%
Appariement par probabilité de dispersion		
Erreur Moyenne	68,7%	60,8%
Ecart Type	1,7%	2,4%

TABLE 7.7 – Classification par graphes aléatoires d'arêtes avec la base GrCor

Nous constatons que le type d'appariement influe peu sur les résultats. L'algorithme de GOLD ET COLL. est toujours plus performant quand l'appariement avec les graphes aléatoires se fait par rapport aux moyennes. Pour l'algorithme de RANGARAJAN ET COLL. la différence entre les deux approches est plus nuancée avec un écart type légèrement plus important.

D'un point de vue général, et cette conclusion vaudra globalement pour les autres bases, les scores sont moins bons avec les graphes d'arêtes. La raison est à chercher dans la combinaison de deux facteurs, la normalité des attributs et la structure.

Regardons d'abord ce qu'il en est de la normalité des attributs aléatoires. Comme pour les graphes primaires, nous nous sommes intéressé au graphe aléatoire 'α'. Les tests de normalité sont donnés dans la table 7.8 page ci-contre. Cette fois-ci les nœuds correspondent aux primitives numérotées en bleu sur la figure 7.1 page 109 et les arêtes représentent les relations d'adjacence entre les nœuds. Rappelons que l'attribut associé aux arêtes est un angle et que le test n'utilise pas la soustraction périodique ce qui peut légèrement biaiser le résultat de normalité.

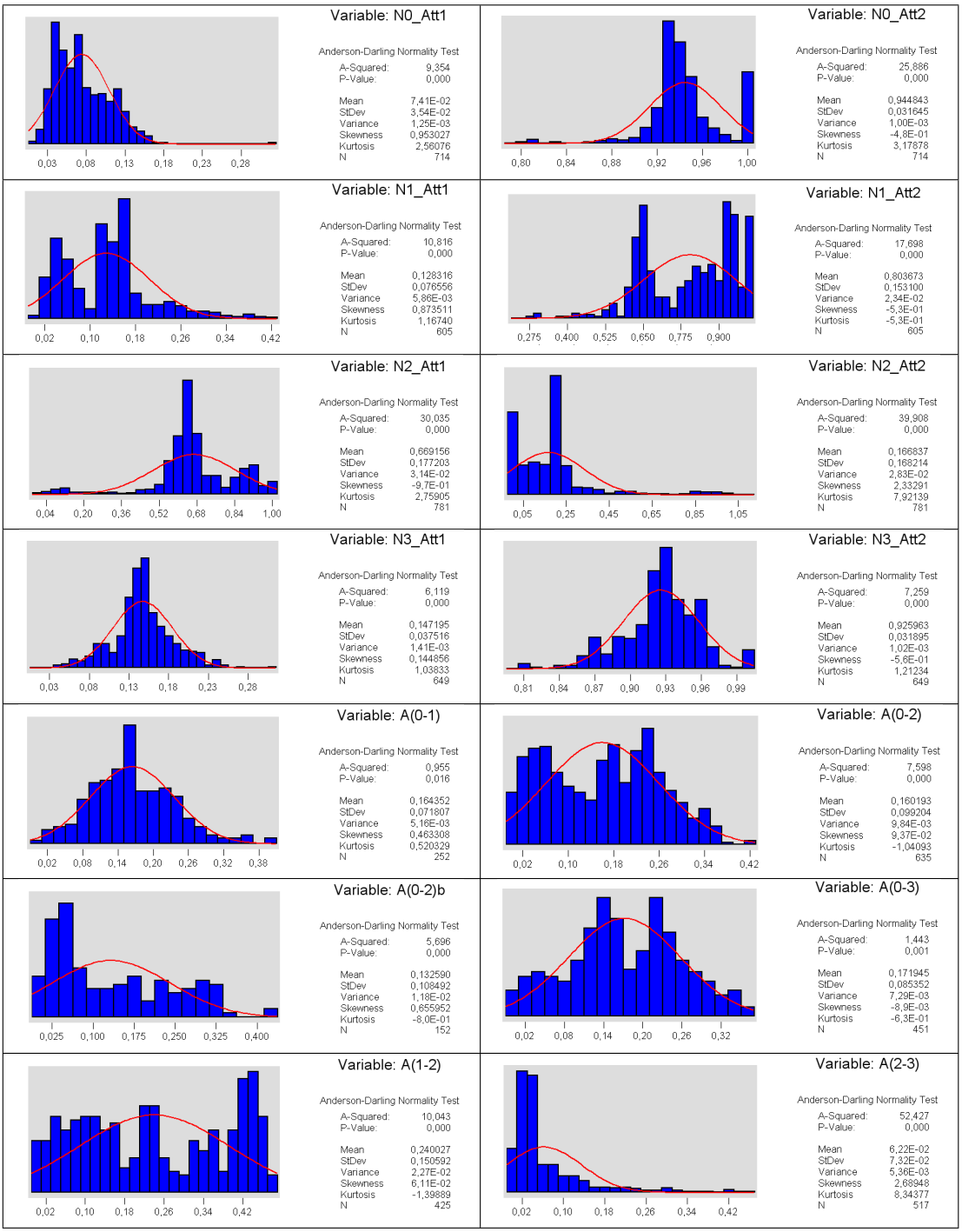


TABLE 7.8 – Tests de normalité réalisés sur les attributs du graphe aléatoire d'arêtes de la classe 'α'. Le nom de la variable de chaque test donne le type d'élément (N pour les nœuds et A pour les arêtes), son numéro dans la structure (comme sur la figure 7.1 page 109 (a)) et le numéro d'attribut pour les nœuds(Att1 $= lr_{ab}$, Att2 $= St_{ab}$) les arêtes n'ayant qu'un seul attribut θ_{ab}.

Pour les graphes d'arêtes comme pour les graphes primaires les attributs ne remplissent jamais la condition de normalité ($p > 0,05$). Le passage en structure d'arêtes n'améliore donc pas la précision de l'appariement.

Si nous combinons cette constatation avec le fait que la structure d'arêtes est plus flexible (nous perdons les informations géométriques liées aux nœuds du graphe primaire), il n'est pas étonnant d'obtenir

des résultats moins performants.

7.1.2.2 Base MNIST

La table 7.9 présente les résultats obtenus avec les graphes d'arêtes dans les mêmes conditions qu'avec les graphes primaires.

Algorithmes d'Appariement	GOLD ET COLL.	RANGARAJAN ET COLL.
Appariement par moyenne		
Erreur Moyenne	61,8%	58,5%
Ecart Type	0,6%	1,0%

TABLE 7.9 – Classification par graphes aléatoires d'arêtes avec la base MNIST

Les remarques faites sur la base GrCor restent valables pour celle-ci pour les mêmes raisons. Les graphes d'arêtes sont également moins discriminants que les graphes primaires et particulièrement pour l'algorithme de GOLD ET COLL. qui introduit moins d'erreurs lors de l'appariement. Les résultats sont ici aussi légèrement moins bons qu'avec les structures primaires.

7.1.2.3 Base GrAnc

Les résultats sur la base GrAnc avec les graphes d'arêtes sont donnés dans la table 7.10.

Algorithmes d'Appariement	GOLD ET COLL.	RANGARAJAN ET COLL.
Appariement par moyenne		
Erreur Moyenne	70,6%	66,1%
Ecart Type	2,6%	3,8%

TABLE 7.10 – Classification par graphes aléatoires d'arêtes avec la base GrAnc

Même constatation que pour les autres bases, les résultats sont nettement moins bons qu'avec les graphes primaires.

7.1.3 Discussion

D'une manière générale les résultats de classification par graphes primaires et par graphes d'arêtes sont moins bons que ceux obtenus avec les classifieurs statistiques de la première partie. Il n'y a rien d'étonnant à cela et les deux raisons de ces contre performances ont déjà été soulevées : les erreurs d'appariement et la variabilité importante qu'ils impliquent sur les attributs aléatoires. Rappelons tout de même qu'aucune optimisation des paramètres d'appariement n'a été effectuée et qu'il conviendrait de mener une étude plus poussée par le biais d'un plan d'expérience pour voir quelles peuvent être les performances maximales atteignables avec ce type d'approche. De plus nous avons remarqué que la complexité des bases, mise en avant en reconnaissance statistique, n'a pas d'influence sur la reconnaissance structurelle proposée. C'est un résultat intéressant qui montre que l'approche structurelle a un sens.

Concernant la différence entre l'algorithme de GOLD ET COLL. et celui de RANGARAJAN ET COLL., les deux approches, nous l'avons vu, sont basées sur le même principe mais utilisent une optimisation différente. Notre conclusion est que celle de GOLD ET COLL. est certainement plus proche de la solution optimale que celle de RANGARAJAN ET COLL..

Enfin pour reprendre les commentaires que nous avons faits sur la comparaison entre les deux types de structures de graphe, il semble bien que la structure primaire soit plus performante dans notre système

que la structure d'arêtes. Cela vient de la structuration plus souple de cette dernière qui est plus sensible aux erreurs d'appariement.

7.2 Coopération

Regardons maintenant quels peuvent être les apports d'une telle approche par rapport aux méthodes statistiques. Malgré les contre performances des approches par graphes aléatoires, la question est de savoir s'ils peuvent apporter une information pertinente dans un système de classification basé sur des approches statistiques. Nous allons, dans cette optique, présenter un schéma de coopération qui nous permettra de répondre à cette question.

7.2.1 Généralités

L'utilisation de systèmes coopérants en reconnaissance de caractères est de plus en plus répandue ([RF03]). Ils reposent sur le principe qu'un ensemble de classifieurs donne bien souvent de meilleurs résultats qu'un classifieur seul. Ceci est d'autant plus vrai que les classifieurs mis en jeu sont indépendants. Il existe de nombreuses méthodes pour rendre les classifieurs indépendants et s'assurer une classification optimale :

- Le « bagging » ([Bre96]), approche compétitive qui sélectionne des ensembles d'apprentissage indépendants pour chaque classifieur. La décision est prise par vote majoritaire.
- Le « boosting » ([Sch90]), approche collaborative où chaque classifieur utilise dans son ensemble d'apprentissage les éléments mal reconnus par le classifieur précédent. La décision est prise par vote majoritaire.
- L'« AdaBoost » ([Sch02]) qui est une version itérative du boosting qui associe à chaque élément de la base un poids. Successivement, chaque classifieur utilise ces poids pour construire son ensemble d'apprentissage puis met les poids à jour en augmentant ceux des caractères qu'il a mal reconnus. Les nouveaux poids sont utilisés par le classifieur suivant et ainsi de suite. La décision est prise par vote majoritaire.

L'utilisation de ces méthodes d'apprentissage est associée à un schéma de reconnaissance dit horizontal (figure 7.2 page suivante). Chaque classifieur ou expert rend son verdict sur la classification d'un caractère inconnu et la décision est prise par un vote majoritaire. Une autre façon de faire coopérer plusieurs experts est d'utiliser une architecture verticale ou « en cascade » ou le n^{e} expert utilise les hypothèses du $n-1^{\text{e}}$ pour réduire son ensemble de possibilités.

7.2.2 Application

7.2.2.1 Choix du schéma de coopération

Au vu des résultats de classification simple, nous proposons de faire coopérer simplement un système statistique reposant sur une description à base de 10 descripteurs de Fourier complexes et de 29 descripteurs de Zernike et une estimation par la règle des k-PPV, avec une classification par graphes aléatoires primaires utilisant l'appariement de GOLD ET COLL. et une similarité par rapport aux moyennes. La coopération s'effectue verticalement suivant le schéma de la figure 7.3 page suivante.

Le rejet est fait en ambiguïté avec un seuil de $0,5$. Ce qui veut dire que si la probabilité *a posteriori* estimée est inférieure à $0,5$ le caractère est rejeté.

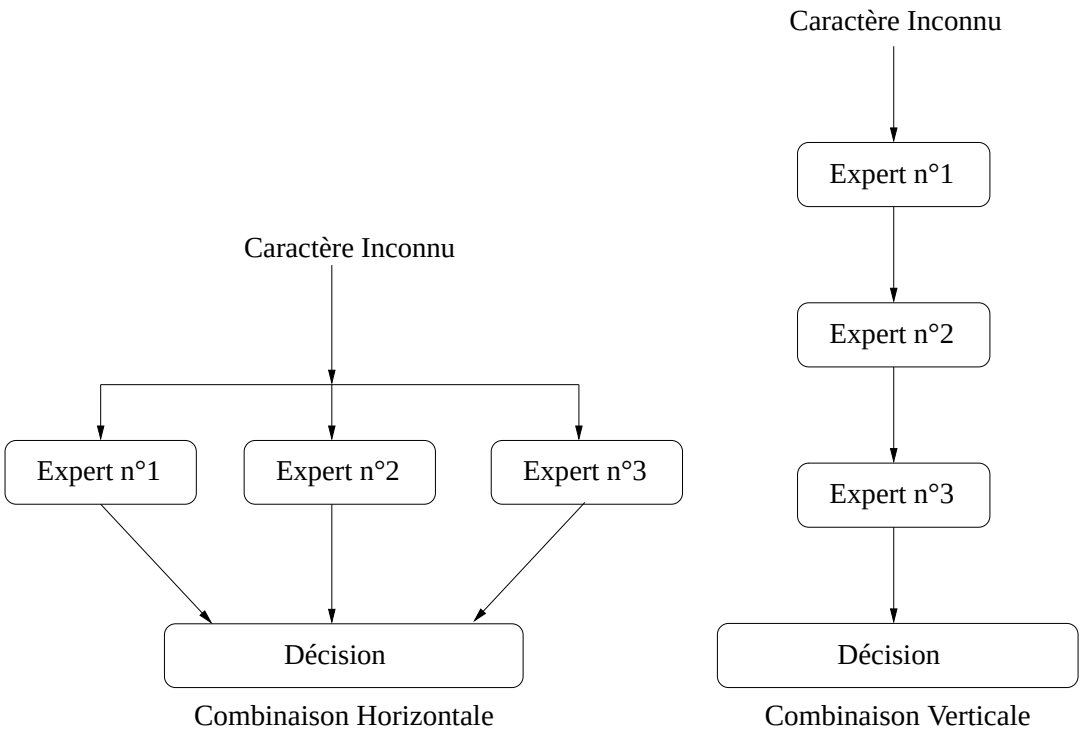


FIGURE 7.2 – Coopération d'experts : les deux architectures classiques.

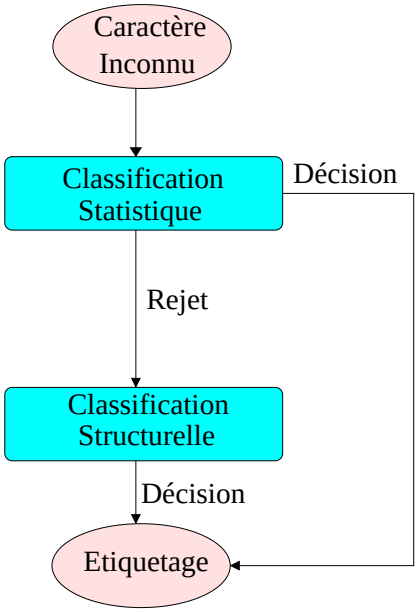


FIGURE 7.3 – Schéma de coopération utilisé.

7.2.2.2 Base GrCor

En reprenant les conditions de tests associées à cette base (quatre groupes de 1 000 individus) nous obtenons les résultats présentés à la table 7.11 page ci-contre.

Nous constatons tout d'abord qu'avec rejet, le classifieur statistique donne les mêmes résultats que sans (voir table 4.4 page 54). Cela valide le choix du seuil de rejet en ambiguïté. La petite amélioration

	Statistique avec rejet	Coopération
Erreur Moyenne	3,6%	3,0%
Ecart Type	0,6%	0,7%

TABLE 7.11 – Résultats de coopération sur la base GrCor.

obtenue sur cette base avec la coopération correspond à une dizaine de caractères ce qui fait qu'environ 20% des caractères rejetés en ambiguïté ont pu être reconnus par la classification structurelle.

7.2.2.3 Base MNIST

Les mêmes tests que précédemment sont présentés à la table 7.12 pour la base MNIST avec quatre groupes de 1 000 individus.

	Statistique avec rejet	Coopération
Erreur Moyenne	27,8%	23,2%
Ecart Type	1,7%	1,3%

TABLE 7.12 – Résultats de coopération sur la base MNIST

Cette fois-ci le rejet en ambiguïté pénalise la reconnaissance statistique puisque nous passons de 22% (table 4.5 page 55) à 28%. La coopération permet à peine de revenir au score du classifieur statistique seul. L'amélioration obtenue correspond à 30% des caractères rejetés en ambiguïté. Les matrices de confusion avant et après la classification structurelle sont données dans les tables 7.13 et 7.14 page suivante respectivement.

0	0	1	2	3	4	5	6	7	8	9	RAm	nbelem
0	**93**	0	0	0	0	0	1	0	1	1	4	369
1	0	**96**	0	0	0	0	0	0	1	0	2	456
2	0	0	**48**	3	7	6	1	2	4	1	28	386
3	0	0	3	**68**	0	1	2	0	6	0	19	412
4	0	0	5	0	**63**	0	1	2	0	8	21	399
5	2	0	6	3	1	**48**	1	4	2	3	30	350
6	0	0	1	1	0	0	**66**	3	1	17	12	395
7	0	1	0	0	1	0	3	**77**	0	3	14	404
8	1	0	0	2	1	0	1	0	**83**	1	9	416
9	0	1	1	1	1	1	8	0	0	**73**	14	413
Pertinence	31	48	3	6,8	5,73	6	3,67	7	5,53	2,15		

TABLE 7.13 – Matrice de confusion (en pourcentage de bonnes reconnaisssances) pour la reconnaissance statistique avec rejet sur la base MNIST. Les colonnes correspondent aux valeurs reconnues, les lignes aux valeurs vraies.

L'amélioration touche surtout les chiffres '2' et '5'. Pour le '5' ceci est cohérant avec les résultats de la classification par graphes aléatoires seule puisque cette classe était assez pertinente. En revanche pour le '2' c'est plus surprenant et montre bien que les deux types d'informations, statistique et structurelle, sont différents et complémentaires.

Les résultats sur les pertinences sont à comparer avec ceux du classifieur statistique seul de la table 4.6 page 56. Les conclusions sont partagées. Nous constatons une amélioration sensible pour les chiffres '3' et '4' qui est intéressante par contre celle-ci se fait au détriment du '0' et du '1' qui étaient relativement bien isolés avec le k-PPV seul. L'augmentation des pertinences pour les chiffres '6', '8' et '9'

0	0	1	2	3	4	5	6	7	8	9	nbelem
0	**96**	0	1	0	0	0	1	0	1	1	369
1	0	**96**	0	0	0	1	0	0	1	0	456
2	5	0	**60**	4	8	8	3	6	4	2	386
3	5	0	4	**74**	0	3	3	4	6	1	412
4	1	1	10	0	**69**	1	4	3	0	11	399
5	3	0	8	8	2	**63**	3	8	2	3	350
6	5	1	1	1	1	3	**67**	4	1	17	395
7	4	1	3	0	1	0	4	**81**	0	5	404
8	4	0	1	3	2	1	2	1	**83**	2	416
9	3	1	3	1	2	6	8	2	0	**74**	413
Pertinence	3,2	24	1,94	4,35	4,31	2,74	2,39	2,89	5,53	1,76	

TABLE 7.14 – Matrice de confusion (en pourcentage de bonnes reconnaisssances) pour la coopération avec la base MNIST. Les colonnes correspondent aux valeurs reconnues, les lignes aux valeurs vraies.

n'est, quant à elle, que la conséquence d'une mauvaise classification de ces caractères avec l'approche par graphes.

7.2.2.4 Base GrAnc

Pour la base GrAnc, les résultats sont présentés dans la table 7.15.

	Statistique avec rejet	Coopération
Erreur Moyenne	15,0%	12,7%
Ecart Type	3,7%	2,1%

TABLE 7.15 – Résultats de coopération sur la base GrAnc

La coopération semble améliorer les résultats et ce même par rapport au k-PPV seul ($13,7\%$ dans la table 4.8 page 56). Cette amélioration reste dans la marge d'incertitude indiquée par l'écart type mais elle correspond à 48% des caractères rejetés en ambiguïté. Ce résultat confirme que l'approche structurelle se comporte bien avec les bases bruitées.

7.2.3 Discussion

Malgré le manque de précision du système de reconnaissance structurelle en l'état, la coopération a un sens et ce d'autant plus que la base est complexe et pose des problèmes aux classifieurs statistiques classiques (base MNIST et GrAnc).

La mise en coopération met en évidence la complémentarité des deux approches de reconnaissance. Il conviendrait maintenant d'optimiser les paramètres d'appariement et le poids de non association du système graphique (le facteur $0,1^{Na}$ sur la probabilité de réalisation voir paragraphe 6.4.3 page 105).

Avec cette approche par graphes aléatoires, nous sommes resté dans le cadre de la reconnaissance avec apprentissage, ce qui limite les applications aux écritures à vocabulaire restreint. Nous allons maintenant nous intéresser à l'utilisation des méthodes structurelles dans le cadre de la reconnaissance générique basée sur la mesure de ressemblance entre une forme et un modèle.

Troisième partie

Les Graphes d'Attributs Hierarchiques Flous (GAHF), une solution graphique pour la reconnaissance de caractères complexes

CHAPITRE 8

PRINCIPE ET MISE EN ŒUVRE

Sommaire

La chaîne de reconnaissance graphique probabiliste décrite dans la deuxième partie de ce mémoire ne peut pas être utilisée dans le cas des écritures à très large vocabulaire avec peu d'éléments par classe. Pour les écritures idéographiques comme le chinois ou les hiéroglyphes égyptiens, le nombre de caractères possible exclut toute tentative d'apprentissage probabiliste. Il faut alors concevoir un système de reconnaissance uniquement basé sur un modèle. C'est l'objet du système de reconnaissance par graphes d'attributs hiérarchiques flous (GAHF) présenté dans ce chapitre.

8.1 Logique floue et descriptions structurelles

La reconnaissance basée sur des modèles pose la question de la modélisation des incertitudes et de l'imprécision. Nous cherchons à prendre en compte la variabilité d'une écriture que nous ne connaissons pas. La première solution serait de la définir *a priori* par rapport à des critères géométriques de l'image du modèle et de connaissances d'experts puis de la raffiner au fur et à mesure que la base grandit. L'efficacité d'une telle approche est dépendante de la diversité du vocabulaire et de la pertinence du choix

originel. La seconde solution serait de quantifier notre ignorance. C'est cette approche que nous utilisons intuitivement quand nous déchiffrons une écriture inconnue : « Ce caractère coréen ressemble à peu près à celui de mon guide ou du moins mieux qu'il ne ressemble aux autres ». Nous venons ici de quantifier notre ignorance par une description linguistique vague « à peu près » et « mieux que ». Malheureusement cette quantification, suffisante pour nous, n'est pas compréhensible par un ordinateur. Il faudrait pouvoir « calculer avec des mots » et c'est précisément ce que propose ZADEH avec le formalisme de la logique floue ([Zad00]).

La logique floue a été introduite par ZADEH ([Zad65]) en 1965 dans le but de décrire l'incertain, l'ambiguïté et le vague. Nous ne reviendrons pas sur son formalisme, les différentes définitions et notations utilisées par la suite sont données en annexe D page 175.

En reconnaissance de caractères, la logique floue, comme dans beaucoup d'autres domaines, est de plus en plus présente. Elle est utilisée dans les systèmes de classification (les systèmes neuro-flous par exemple [GGD+98]) et de décision (les arbres de décision flous [ZCSY03]). Quelques systèmes basés sur des descriptions structurelles floues ont également vu le jour pour des applications spécifiques. Deux approches sont à ce titre particulièrement intéressantes, les descriptions syntaxiques floues avec les travaux de MALAVIYA ET PETERS ([MP00]) autour du projet FOHDEL[1] et les graphes d'attributs flous introduits par CHAN ET CHEUNG ([CC92]).

8.1.1 Les langages flous, l'exemple de FOHDEL

Les langages flous sont dérivés des langages syntaxiques ([GT78]). Un langage est un ensemble fini ou infini et dénombrable de phrases construites à partir d'un alphabet. Les règles de construction de ces phrases ainsi que l'alphabet constituent la grammaire du langage. Un langage flou repose sur une grammaire floue qui est une grammaire pour laquelle les règles de construction sont associées à une fonction d'appartenance :

Définition 32 (Grammaire floue) *Une grammaire floue $\tilde{\mathcal{G}}$ est un 4-uplet $\tilde{\mathcal{G}} = (V_n, V_t, P, S)$ où :*
- *V_n est un ensemble fini de* non-terminaux *ou variables,*
- *V_t est un ensemble fini de* terminaux *ou constantes,*
- *P est un ensemble fini de* règles de production *du type $\{\alpha \xrightarrow{\mu} \beta\}$, où α et β sont des éléments de $V = V_n \cup V_t$ représentant l'*alphabet *de $\tilde{\mathcal{G}}$ et $\mu \in [0, 1]$ est la fonction d'appartenance de la règle de production,*
- *$S \in V_n$ est le symbole de départ.*

Le langage flou $\mathcal{L}(\tilde{\mathcal{G}})$ se définit alors de la manière suivante :

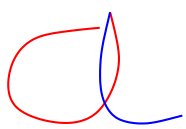
$$\mathcal{L}(\tilde{\mathcal{G}}) = \{(x, \mu(x)) | x \in V_t^*, S \overset{*}{\Rightarrow} x\} \tag{8.1}$$

où V_t^* désigne la fermeture de V_t c'est-à-dire l'ensemble dénombrable de toutes les phrases dérivant de V_t et le signe $\overset{*}{\Rightarrow}$ indique l'utilisation de zéro, ou plus, règles de production de P. Quant à μ, elle peut être obtenue par une t-norme (annexe D page 175) sur toutes les règles de production utilisées.

FOHDEL ([MP00]) est un langage flou dont l'alphabet est composé de primitives sémantiques, c'est-à-dire des descriptions linguistiques de la forme, de la taille et de la position des segments primitifs des caractères. L'originalité de ce langage est qu'il ajoute à l'alphabet des attributs de quantification (« Z, L, H, … » pour « Zero, Large, High, … ») et des opérateurs flous permettant d'associer ou de comparer deux primitives. Ainsi, il inclut non seulement les facteurs d'incertitude sur les règles de production d'une phrase mais également sur les éléments de la phrase. Chaque classe de caractères est modélisée par

[1] Fuzzy On-line Handwriting DEscription Language

une grammaire floue dont l'alphabet et la base de règles sont déterminés automatiquement par apprentissage. Pour la phase de classification le caractère inconnu est décomposé puis interprété sous forme d'une règle (un exemple de règle est donné à la figure 8.1). Les degrés d'appartenance du caractère à la règle sont ensuite comparés à ceux obtenus par chaque grammaire. La décision se base sur cette comparaison.



Très grand degré d'appartenance à une primitive « en forme de O » ou « en forme de C » à gauche et grand degré d'appartenance à une primitive « petit crochet » sur la droite

règle a : $(VH\#O\_L \mid VH\#C\_L)$&
$(H\#HOR\_R \mid H\#NS\_MR)$

FIGURE 8.1 – Exemple de règle de construction d'un « a » avec FOHDEL.

Ce langage est relativement complexe dans sa mise en œuvre et basé sur un système d'extraction de primitives développé pour les écritures en-ligne ([Mal96]). La structure du caractère est unidimensionnelle puisque les relations n'existent qu'entre deux primitives connexes dans l'écriture. L'adaptation aux caractères manuscrits complexes comme les hiéroglyphes égyptiens passe par l'utilisation d'une structure bi-dimensionnelle qui complexifie énormément la syntaxe du langage.

8.1.2 Les graphes d'attributs flous de CHAN ET CHEUNG

Plus adaptés à l'écriture manuscrite hors-ligne, les graphes d'attributs flous de CHAN ET CHEUNG ([CC92]) sont une extension des graphes d'attributs de FU ([Fu82]).

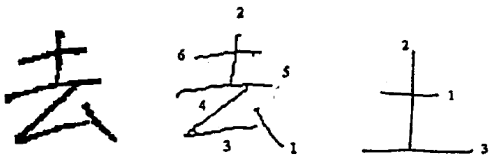
Chaque nœud d'un graphe d'attributs flous prend ses attributs dans un ensemble $Z = \{z_i | i = 1, ..., I\}$ où chaque attribut z_i va prendre ses valeurs dans l'ensemble $S_i = \{s_{ij} | j = 1, ..., J_i\}$. L'ensemble $\tilde{L_v} = \{(z_i, \tilde{A_{S_i}}) | i = 1, ..., I\}$ est alors l'ensemble des couples de valeurs possibles pour les primitives où $\tilde{A_{S_i}}$ est un ensemble flou sur l'ensemble des valeurs S_i. Une primitive valide est un sous-ensemble de $\tilde{L_v}$ dans lequel chaque attribut ne peut apparaître qu'une seule fois. L'ensemble de toutes ces primitives valides est noté $\tilde{\Pi}$ et chaque nœud est représenté par un élément de $\tilde{\Pi}$.

De manière similaire, pour les arêtes, l'ensemble des attributs possibles est appelé $F = \{f_i | i = 1, ..., I'\}$ dans lequel chaque f_i peut prendre ses valeurs dans $T_i = \{t_{ij} | j = 1, ..., J'_i\}$. L'ensemble $\tilde{L_a} = \{(f_i, \tilde{B_{T_i}}) | i = 1, ..., I'\}$ est l'ensemble des couples de valeurs possibles pour les relations où $\tilde{B_{T_i}}$ est un ensemble flou sur l'ensemble des valeurs T_i. Une relation valide est alors un sous-ensemble de $\tilde{L_a}$ dans lequel chaque attribut ne peut apparaître qu'une seule fois. L'ensemble de toutes les relations valides est noté $\tilde{\Theta}$.

En repartant de la définition des graphes d'attributs donné au paragraphe 6.2.1 page 90, les graphes d'attributs flous se définissent formellement de la manière suivante :

Définition 33 (Graphe d'attributs flous ou GAF) *un graphe d'attributs flous* $\tilde{Ga}$ *sur* $\tilde{L} = \{\tilde{L_v}, \tilde{L_a}\}$ *avec une structure graphique* $G = (N, A)$*, est une paire ordonnée* $(\tilde{V}, \tilde{E})$ *où* $\tilde{V} = (\tilde{N}, \tilde{\sigma})$ *est appelé un* ensemble de nœuds flous *et* $\tilde{E} = (\tilde{A}, \tilde{\delta})$ *est appelé un* ensemble d'arêtes floues*. Les applications* $\tilde{\sigma} : \tilde{N} \rightarrow \tilde{\Pi}$ *et* $\tilde{\delta} : \tilde{A} \rightarrow \tilde{\Theta}$ *sont appelées respectivement* interpréteurs *de nœuds flous et d'arêtes floues.*

CHAN ET CHEUNG utilisent ce formalisme pour la description de caractères manuscrits chinois (un exemple est donné à la figure 8.2 page suivante (a)). Ces derniers sont décomposés en segments droits qui constituent les nœuds du graphe et les arêtes correspondent aux relations d'adjacence et de position entre segments (figure 8.2 page suivante (b)). Les attributs utilisés pour les nœuds décrivent l'orientation du segment (horizontal, vertical, orienté à 45 °ou à 135 °), ceux des arêtes décrivent le type de connection



(a) (b) (c)

FIGURE 8.2 – Exemple de caractère traité par CHAN ET CHEUNG ([CC92]) (a) avant amincissement, (b) après amincissement, (c) exemple de radical à apparier.

(les segments reliés se coupent en « $\times$ », en « $\perp$ », en « $\top$ », en « $\vdash$ » ou en « $\dashv$ » sont parallèles ou non) et les positions relatives (en haut, en bas, à gauche, à droite ou sans relation). Ils réalisent ensuite un appariement par rapport à des radicaux modèles (figure 8.2 (c)) décrits sous forme de graphes d'attributs non flous (c.-à-d. avec des fonctions d'appartenance valant 0 ou 1) en utilisant les mesures de similarités entre nœuds (notée $\alpha_1(v_1, v_2)$) et entre arêtes (notée $\alpha_2(a_1, a_2)$) suivantes :

$$\alpha_1(v_1, v_2) = \bigwedge_{i=1}^{I} \bigvee_{j=1}^{J_i} (\mu_{\tilde{A_{1S_i}}}(s_{ij}) \wedge \mu_{\tilde{A_{2S_i}}}(s_{ij})) \tag{8.2}$$

pour les nœuds et

$$\alpha_2(a_1, a_2) = \bigwedge_{i=1}^{I'} \bigvee_{j=1}^{J'_i} (\mu_{\tilde{B_{1T_i}}}(t_{ij}) \wedge \mu_{\tilde{B_{2T_i}}}(t_{ij})) \tag{8.3}$$

pour les arêtes, où $\mu_{\tilde{A_{KS_i}}}$ représente la fonction d'appartenance de s_{ij} à l'ensemble flou $\tilde{A_{KS_i}}$, $\mu_{\tilde{B_{KT_i}}}$ représente la fonction d'appartenance de t_{ij} à l'ensemble flou $\tilde{B_{KS_i}}$, $K = 1, 2$ et $\wedge$ et $\vee$ sont les opérateurs min et max.

Ces mesures sont volontairement très restrictives puisqu'elles mesurent la ressemblance entre attributs flous et attributs binaires. L'appariement par rapport aux radicaux se fait par maximisation de la similarité entre graphes définie par :

$$\gamma(\tilde{G_1}, \tilde{G_2}) = \bigwedge_{i \in N_1} \alpha_1(i, h(i)) \times \bigwedge_{(i,j) \in E_1} \alpha_2(a_1(i, j), a_2(h(i), h(j))) \tag{8.4}$$

où $h(i)$ est le nœud de $\tilde{G_2}$ qui est apparié au nœud i de $\tilde{G_1}$ et $a_k(i, j)$ est l'arête reliant les nœuds i et j dans $\tilde{G_k}$. La reconnaissance se fait ensuite par rapport aux radicaux qui sont contenus dans le caractère à reconnaître.

La nature des caractères chinois (pouvant se décomposer en segments droits) s'adapte relativement bien à cette approche moyennant un algorithme d'extraction des primitives efficace. Le nombre d'attributs est relativement restreint et les radicaux parfaitements définis. Malheureusement ces deux conditions ne peuvent pas être remplies dans le cas de caractères manuscrits complexes comme les hiéroglyphes égyptiens.

8.2 Vers une description complète, les GAHF

La reconnaissance à partir des graphes d'attributs hiérarchiques flous (GAHF) proposée ici part de la constatation qu'un système de reconnaissance générique de caractères pouvant traiter des caractères simples comme complexes ne peut pas reposer sur une décomposition en radicaux binaires. Pour les

mêmes raisons qui nous ont poussé à écarter le choix du graphe médian généralisé de JIANG ET COLL. ([JMB01]) comme graphe modèle d'une classe dans le cas des graphes d'attributs, la recherche de radicaux par apprentissage est coûteuse et risque d'introduire une erreur supplémentaire due à l'appariement. Nous avons donc gardé notre idée première du choix du graphe d'ordre minimal comme modèle d'une classe.

A partir de la décomposition des caractères en segments primitifs, explicitée dans le chapitre 5 page 61, nous proposons ici une approche générique de décomposition d'un caractère en primitives sémantiques adaptable à une forme quelconque. Cette décomposition fait intervenir une hiérarchie dans les attributs qui peut être modélisée par des arbres flous. Nous avons ensuite défini une extension des graphes d'attributs flous pour la phase de reconnaissance.

8.2.1 De l'intérêt d'une hiérarchie linguistique

Le principe de la reconnaissance par GAHF est de décrire la forme directement à partir des segments primitifs à convexité stricte qui représentent le niveau sémantique le plus élevé disponible pour un système générique. La notion de convexité stricte est importante puisqu'elle simplifie l'alphabet de la grammaire à utiliser pour la description.

L'introduction de la hiérarchie dans la description linguistique des primitives est justifiée par la volonté de pouvoir affiner la description en fonction de la base et de l'utilisateur. Prenons l'exemple de la figure 8.3 (a). L'attribut « Type » associé à la description de la forme des primitives 1 et 2 pour les objets A et B peut prendre ses valeurs dans $A_{Type} = \{Cercle, Ellipse\}$ ou pour une description plus précise $A'_{Type} = \{Cercle, Ellipse\_Gauche, Ellipse\_Droite\}$. Or une comparaison des objets effectuée sur ce critère donnerait dans les deux cas des résultats complètement différents. Avec A_{Type} les objets seraient trouvés identiques alors qu'avec A'_{Type} ils seraient différenciés. Maintenant si cette question est posée à l'utilisateur du sytème la réponse variera suivant l'application et la précision recherchée. Distinguer un « 6 » d'un « 9 » est indispensable pour un système OCR. En revanche, vouloir apporter une description fine des caractères hiéroglyphes pour distinguer une position de bras sur un caractère humain au risque que le système ne retourne aucune réponse, peut ne pas convenir à l'utilisateur. Une solution consiste à utiliser une description par arbre permettant de choisir le niveau de précision souhaité. L'arbre de description donné en (b) pour l'exemple de la figure 8.3 (a) permet d'adapter la décision. A la profondeur '0' l'attribut n'est pas considéré, à la profondeur '1' l'attribut prend ses valeurs dans A_{Type} et à la profondeur '2' il utilise A'_{Type}.

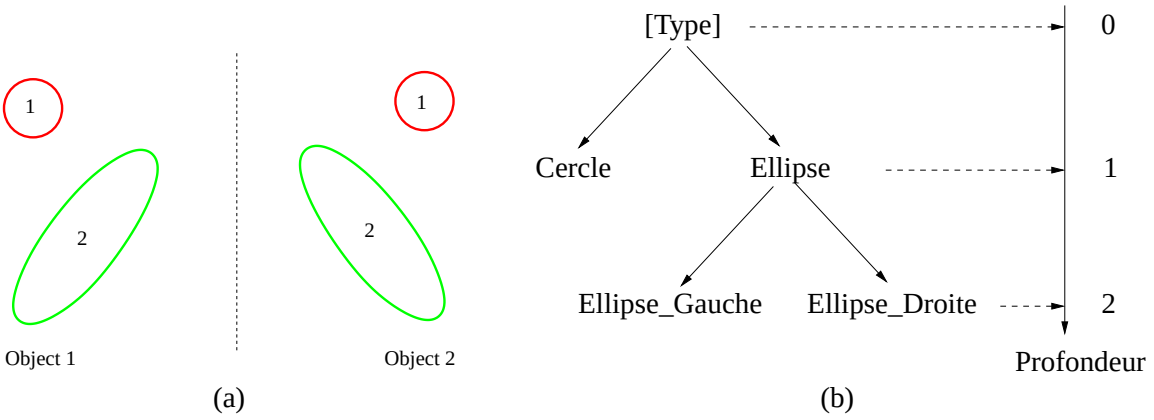


FIGURE 8.3 – Illustration de l'intérêt d'une description linguistique hiérarchique, (a) objets à apparier, (b) arbre linguistique pour le type de forme

8.2.2 Définition

Les GAHF sont des graphes d'attributs flous au sens de CHAN ET CHEUNG dont les attributs sont des arbres linguistiques flous. Nous avons déjà parlé des arbres comme étant des forêts connectées au paragraphe 5.3.2.2 page 82, une autre définition plus classique est donnée par GONZALES ET THOMASON ([GT78]) :

Définition 34 (Arbres) *Un arbre t est un ensemble de nœuds tel que :*
- *un nœud unique est appelé* racine *de t,*
- *les autres nœuds sont divisés en ensembles disjoints $t_1, ..., t_n$ où t_i est appelé un sous-arbre de t.*

D'un point de vue informatique, un nœud x est donc une structure contenant un pointeur sur son prédécesseur (nul si le nœud est une racine) et une liste de pointeurs sur ses successeurs (vide si le nœud est une feuille). Il lui est associé également un entier appelé *rang* (noté $r(x)$), une étiquette appelée *index* (x dans le cas présent) et un entier appelé *profondeur* (noté $h(x)$). Le rang est le nombre de successeurs du nœud (il vaut zéro pour une feuille). L'index est une chaîne de caractères définie par concaténation par rapport à l'index de son prédécesseur. Il vaut $x = k.p$ si k est l'index du prédécesseur (zéro s'il s'agit d'une racine) et si p est la place qu'occupe le nœud dans la liste des successeurs de son prédécesseur (numérotés de gauche à droite par convention). La profondeur, quant à elle, est le niveau du nœud dans l'arbre et se définit de manière récursive :
- $h(x) = 1$, si $x = 0$;
- $h(x.n) = h(x) + 1$.

Définition 35 (Arbres flous) *Un arbre flou est un arbre dont les nœuds sont associés à un degré d'appartenance $\mu \in [0, 1]$. Il est noté $\tilde{t} = (t, \mu)$ où t est un arbre.*

Nous appelons arbres linguistiques flous des arbres flous pour lesquels tous les successeurs d'un nœud représentent la valeur d'une variable linguistique[2].

Les GAHF se définissent de la même manière que les graphes d'attributs flous. Les nœuds d'un GAHF prennent leurs attributs hiérarchiques dans l'ensemble $Z = \{z_i | i = 1, ..., I\}$ où chaque z_i prend ses valeurs dans l'ensemble d'arbres $\hat{S}_i = \{\hat{s_{ij}} | j = 1, ..., J_i\}$. L'ensemble $\hat{L_v} = \{(z_i, \hat{A_{S_i}}) | i = 1, ..., I\}$ représente l'ensemble des couples de valeurs possibles pour les primitives avec $\hat{A_{S_i}} = \{(\hat{s_{ij}}, \mu_j) | j = 1, ..., J_i\}$ l'ensemble des arbres linguistiques flous qui peuvent être associés à z_i. Comme pour les graphes d'attributs flous, une primitive valide est un sous-ensemble de $\hat{L_v}$ dans lequel chaque attribut ne peut apparaître qu'une seule fois. L'ensemble de toutes ces primitives valides est noté $\hat{\Pi}$ et chaque nœud est représenté par un élément de $\hat{\Pi}$.

Pour les arêtes, l'ensemble des attributs possibles est $F = \{f_i | i = 1, ..., I'\}$ dans lequel chaque f_i peut prendre ses valeurs dans l'ensemble d'arbres $\hat{T_i} = \{\hat{t_{ij}} | j = 1, ..., J'_i\}$. L'ensemble $\hat{L_a} = \{(f_i, \hat{B_{T_i}}) | i = 1, ..., I'\}$ et $\hat{B_{T_i}} = \{(\hat{t_{ij}}, \mu_j) | j = 1, ..., J_i\}$ l'ensemble des arbres linguistiques flous qui peuvent être associés à f_i. Une relation valide est un sous-ensemble de $\hat{L_a}$ dans lequel chaque attribut ne peut apparaître qu'une seule fois. L'ensemble de toutes les relations valides est noté $\hat{\Theta}$. Et enfin :

Définition 36 (Graphe d'attributs hiérarchiques flous ou GAHF) *un graphe d'attributs hiérarchiques flous $\hat{G}a$ sur $\hat{L} = \{\hat{L_v}, \hat{L_a}\}$ avec une structure graphique $G = (N, A)$, est une paire ordonnée $(\hat{V}, \hat{E})$ où $\hat{V} = (\hat{N}, \hat{\sigma})$ est appelé un* ensemble de nœuds hiérarchiques flous *et $\hat{E} = (\hat{A}, \hat{\delta})$ est appelé un* ensemble d'arêtes hiérarchiques floues*. Les applications $\hat{\sigma} : \hat{N} \rightarrow \hat{\Pi}$ et $\hat{\delta} : \hat{A} \rightarrow \hat{\Theta}$ sont appelées respectivement* interpréteurs *de nœuds hiérarchiques flous et d'arêtes hiérarchiques floues.*

[2] La définition d'une variable linguistique est redonnée en annexe D.2.1 page 180

Ces notations donnent lieu à deux remarques. Tout d'abord, notons que si les arbres linguistiques flous associés aux attributs sont de profondeur '1', nous revenons au formalisme classique des graphes d'attributs flous de CHAN ET CHEUNG à une nuance près. Les $\hat{s_{ij}}$ et les $\hat{t_{ij}}$ ne représentent plus alors une valeur possible de l'attribut mais un ensemble de valeurs qui, associées aux μ_j, forment des ensembles de sous-ensembles flous représentés par $\hat{A_{S_i}}$ ou $\hat{B_{T_i}}$. La seconde remarque concerne la notion d'arbre linguistique flou. Celle-ci peut paraître redondante puisqu'il ne s'agit finalement que d'arbres flous avec une interprétation particulière. Cependant cette interprétation permet, dans le cas où la profondeur est égale à 1, de pouvoir interpréter les $\hat{A_{S_i}}$ $\hat{B_{T_i}}$ comme étant les ensembles caractéristiques d'une variable linguistique. Pour les profondeurs supérieures à 1, elle permet également de définir proprement les mesures de similarité entre éléments du graphe.

Une autre façon d'interpréter les $\hat{A_{S_i}}$ $\hat{B_{T_i}}$ est de les mettre en parallèle avec les langages d'arbres flous. Une grammaire d'arbres flous est une grammaire floue dont l'alphabet est constitué d'arbres :

Définition 37 (Grammaire d'arbres flous) *Une grammaire d'arbres flous est un 4-uplet* $\tilde{\mathcal{G}}_t = (V_N, V_T, P, S)$*, tel que :*

- V_n *est un ensemble fini de* non-terminaux *ou variables,*
- V_t *est un ensemble fini de* terminaux *ou constantes,*
- P *est un ensemble fini de* règles de production *du type :* $\tau_i \xrightarrow{\mu} \tau_j$*, où* τ_i, τ_j *sont des arbres éléments de* $V = V_n \cup V_t$ *représentant l'*alphabet *de* $\tilde{\mathcal{G}}_t$ *et* $\mu \in [0, 1]$ *est appelé le degré d'appartenance de la production.*
- $S \in V_N$ *est le symbole de départ.*

Un langage d'arbres flous est généré par une grammaire d'arbres flous et se note $\mathcal{L}(\tilde{\mathcal{G}}_t)$. Il se définit de la même manière que pour une grammaire floue (équation 8.1 page 122) :

$$\mathcal{L}(\tilde{\mathcal{G}}_t) = \{(t, \mu) | t \in V_t^*, S \overset{*}{\Rightarrow} t\} \tag{8.5}$$

Il apparaît donc que les phrases d'un langage d'arbres flous sont des arbres flous. Les $\hat{A_{S_i}}$ $\hat{B_{T_i}}$ peuvent ainsi être considérés comme des langages d'arbres flous générés par des grammaires d'arbres flous. SHU ([Shu00]) explicite d'ailleurs un algorithme d'inférence pour les grammaires d'arbres flous à partir d'un ensemble d'arbres flous. Ce formalisme peut être intéressant, par exemple, pour réaliser un apprentissage à partir d'un ensemble de GAHF. Nous n'avons pas abordé cet aspect dans notre système.

8.2.3 Principe de construction

Le principe de construction d'un GAHF repose sur le même principe que les graphes d'attributs présentés dans la partie précédente. Le caractère est squelettisé avec l'algorithme de ZHANG ET WANG, les points singuliers sont extraits et le graphe est construit à partir d'une structure d'arêtes. Les segments primitifs, associés aux nœuds, sont décrits par deux attributs hiérarchiques flous représentant la longueur et le type du segment. Les arêtes sont associées elles aussi à deux attributs hiérarchiques flous représentant la position relative et la connexité. Nous reviendrons dans le prochain paragraphe sur la construction des attributs.

Faisons deux remarques sur les attributs hiérarchiques liés aux arêtes. L'attribut de position relative oblige à orienter les arêtes. En effet si un segment est « à droite » d'un autre, la description inverse, c'est-à-dire « à gauche », est différente et ne peut pas être portée par la même arête. Nous avons donc opté pour une description doublement orientée. De plus comme nous voulons décrire les relations d'adjacence de manière floue, nous avons utilisé une description complète dans laquelle l'adjacence est un attribut de l'arête (il s'agit de ce que nous avons appelé la connexité). Ainsi chaque couple de nœuds du graphe est relié par deux arcs orientés en sens inverses (figure 8.4 page suivante).

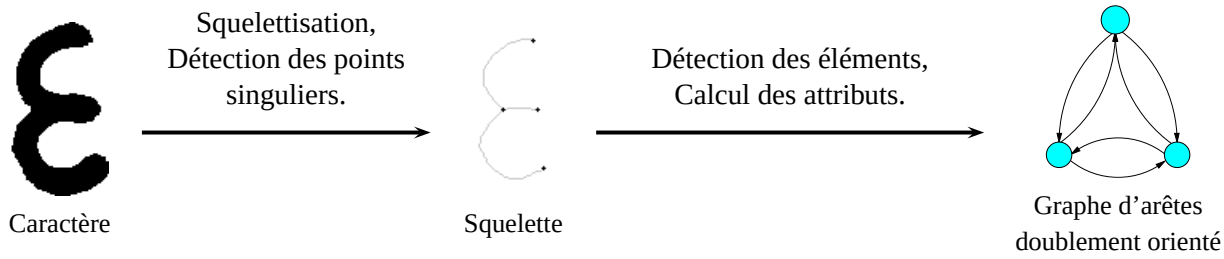


FIGURE 8.4 – Construction d'un graphe complet doublement orienté à partir d'un caractère

Les GAHF peuvent également être construits à partir d'une structure primaire. Ce sont alors les positions des points singuliers qui deviennent floues. Cette description comporte deux inconvénients, premièrement le niveau de flou sur la position des points singuliers est un problème complexe et très dépendant du type de caractères traités, ensuite nous perdons la possibilité de flou sur l'adjacence des segments. C'est pourquoi nous nous sommes focalisé sur les descriptions d'arêtes doublement orientées malgré la complexité combinatoire engendrée. C'est un problème ouvert qui nous limite aujourd'hui pour la reconnaissance de mots trop complexes par exemple. Nous avons pris le parti de décrire complètement et proprement ces structures, leur optimisation n'entre pas, aujourd'hui, dans le cadre de notre travail.

8.2.4 Construction des attributs hiérarchiques

8.2.4.1 Fonctions de « fuzzification »

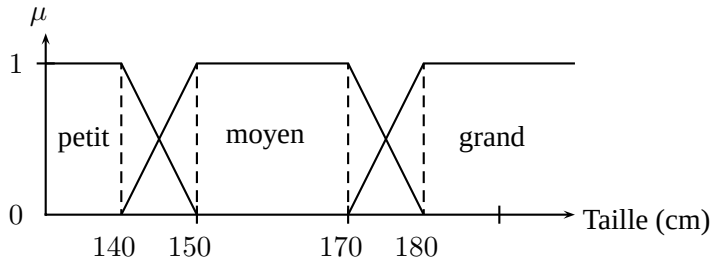


FIGURE 8.5 – Variable linguistique (Taille, $\mathbb{R}^+$, {« grand », « moyen », « petit »})

La construction des attributs passe par le calcul des degrés d'appartenance de l'élément aux sous-ensembles flous caractéristiques d'une variable linguistique. C'est la phase de « fuzzification » ou de granularisation de l'espace. Une manière classique de réaliser cette étape est de partitionner l'ensemble de définition d'un attribut numérique quantifiant la variable linguistique. La figure 8.5 donne l'exemple d'une partition de $\mathbb{R}^+$, ensemble de définition de la mesure d'une taille humaine associée à la variable linguistique « Taille ». Cette partition utilise des fonctions trapézoïdales mais il existe d'autres fonctions classiques pour la fuzzification. Les définitions des trois fonctions les plus utilisées ([MP97]) pour le calcul des degrés d'appartenance sont décrites par la suite. La plus courante est la fonction triangle (figure 8.6 page suivante (a)) qui se définit de la façon suivante :

$$\Lambda(x;b,c) = \begin{cases} 1 - 2\left|\frac{x-c}{b}\right| & \text{si } (c-\frac{b}{2}) \leq x \leq (c+\frac{b}{2}), \\ 0 \text{ sinon.} \end{cases} \tag{8.6}$$

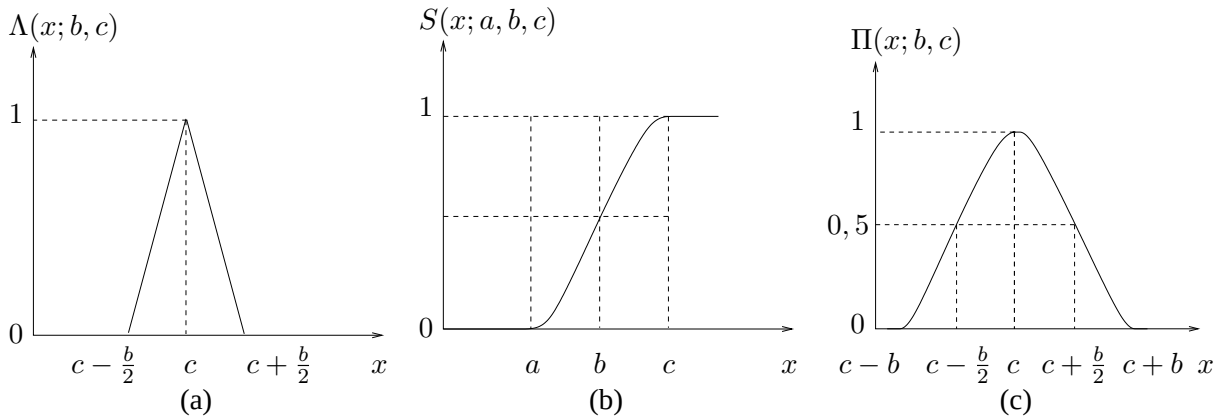


FIGURE 8.6 – Fonctions de fuzzification, (a) fonction Λ, (b) fonction S, (c) fonction Π.

La fonction triangle est facile à programmer mais présente l'inconvénient de partitionner l'espace de façon un peu trop abrupte. Les fonctions en S et en Π (figure 8.6 (b) et (c)) permettent d'obtenir un découpage plus doux :

$$S(x;a,b,c) = \begin{cases} 0 \text{ pour } x \leq a \\ 2\left(\frac{x-a}{c-a}\right)^2 \text{ si } a \leq x \leq b \\ 1 - 2\left(\frac{x-c}{c-a}\right)^2 \text{ si } b \leq x \leq c \\ 1 \text{ si } x \geq c \end{cases} \tag{8.7}$$

$$\Pi(x;b,c) = \begin{cases} S(x;c-b,c-\frac{b}{2},c) \text{ si } x \leq c \\ 1 - S(x;c,c+\frac{b}{2},c+b) \text{ si } x \geq c \end{cases} \tag{8.8}$$

Les attributs hiérarchiques, que nous exposons par la suite, sont principalement construits par fuzzification de grandeurs numériques. Ces dernières sont soit des grandeurs classiques soit des grandeurs que nous avons définies spécifiquement pour notre application. Tous les coefficients des courbes de fuzzification ont été choisis et/ou validés empiriquement sur des exemples simples.

8.2.4.2 Attributs des nœuds

Un nœud représente un segment primitif de convexité stricte. Il supporte deux types d'information : la longueur relativement à la longueur totale du squelette et la forme. La longueur est représentée par l'attribut hiérarchique N_LONG et la forme par l'attribut hiérarchique N_TYPE.

N_LONG : La longueur étant une variable linguistique simple, N_LONG est un arbre linguistique flou de profondeur 1 (figure 8.7 page suivante). Il est obtenu par fuzzification de l'attribut de longueur relative de la primitive reliant les points A et B, lr_{AB}, défini par l'équation 6.1 page 92. La variable linguistique de longueur relative est définie sur $[0,1]$ et peut prendre neuf valeurs {Zero, Très Très Petite, Très Petite, Petite, Moyenne, Grande, Très Grande, Très Très Grande, Excellente } afin d'obtenir une granularité relativement fine. La fuzzification est effectuée suivant le modèle développé dans ([MP97]). Elle utilise des fonctions Π table 8.1 page suivante et figure 8.8 page suivante).

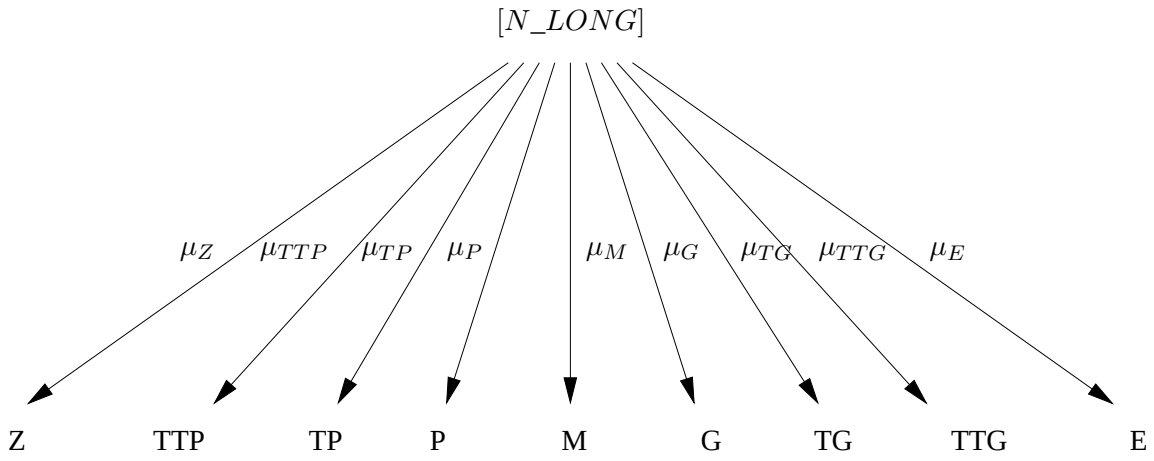


FIGURE 8.7 – Représentation de N_LONG.

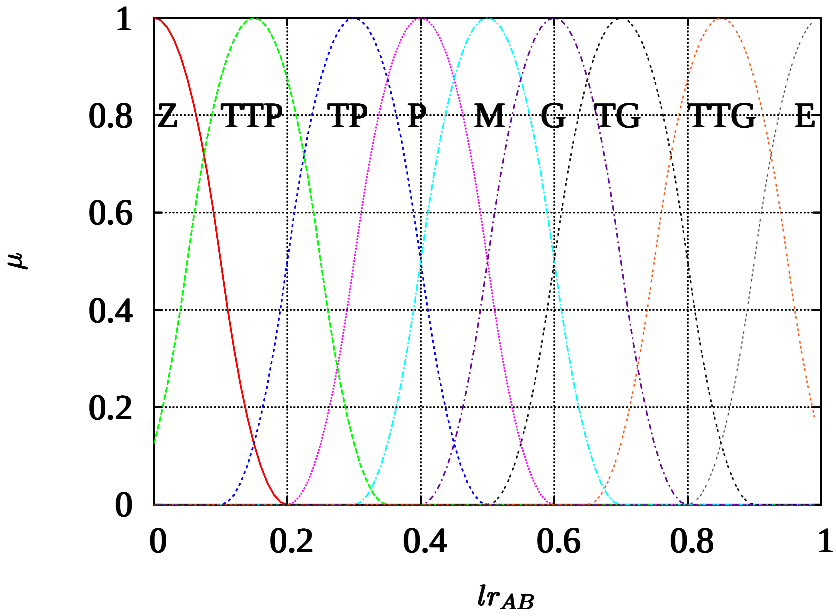


FIGURE 8.8 – Fonctions d'appartenance de la variable linguistique associée à lr_{AB}.

Valeur	Symbole	Paramètres	Fonction
Zero	Z	$b = 0,3\,;\, c = 0$	$\mu_Z(x) = \Pi(x; b, c)$
Très Très Petite	TTP	$b = 0,3\,;\, c = 0,15$	$\mu_{TTP}(x) = \Pi(x; b, c)$
Très Petite	TP	$b = 0,3\,;\, c = 0,3$	$\mu_{TP}(x) = \Pi(x; b, c)$
Petite	P	$b = 0,3\,;\, c = 0,4$	$\mu_P(x) = \Pi(x; b, c)$
Moyen	M	$b = 0,3\,;\, c = 0,5$	$\mu_M(x) = \Pi(x; b, c)$
Grande	G	$b = 0,3\,;\, c = 0,6$	$\mu_G(x) = \Pi(x; b, c)$
Très Grande	TG	$b = 0,3\,;\, c = 0,7$	$\mu_{TG}(x) = \Pi(x; b, c)$
Très Très Grande	TTG	$b = 0,3\,;\, c = 0,85$	$\mu_{TTG}(x) = \Pi(x; b, c)$
Excellente	E	$b = 0,3\,;\, c = 1$	$\mu_E(x) = \Pi(x; b, c)$

TABLE 8.1 – Fonctions d'appartenance de la variable linguistique associée à lr_{AB}.

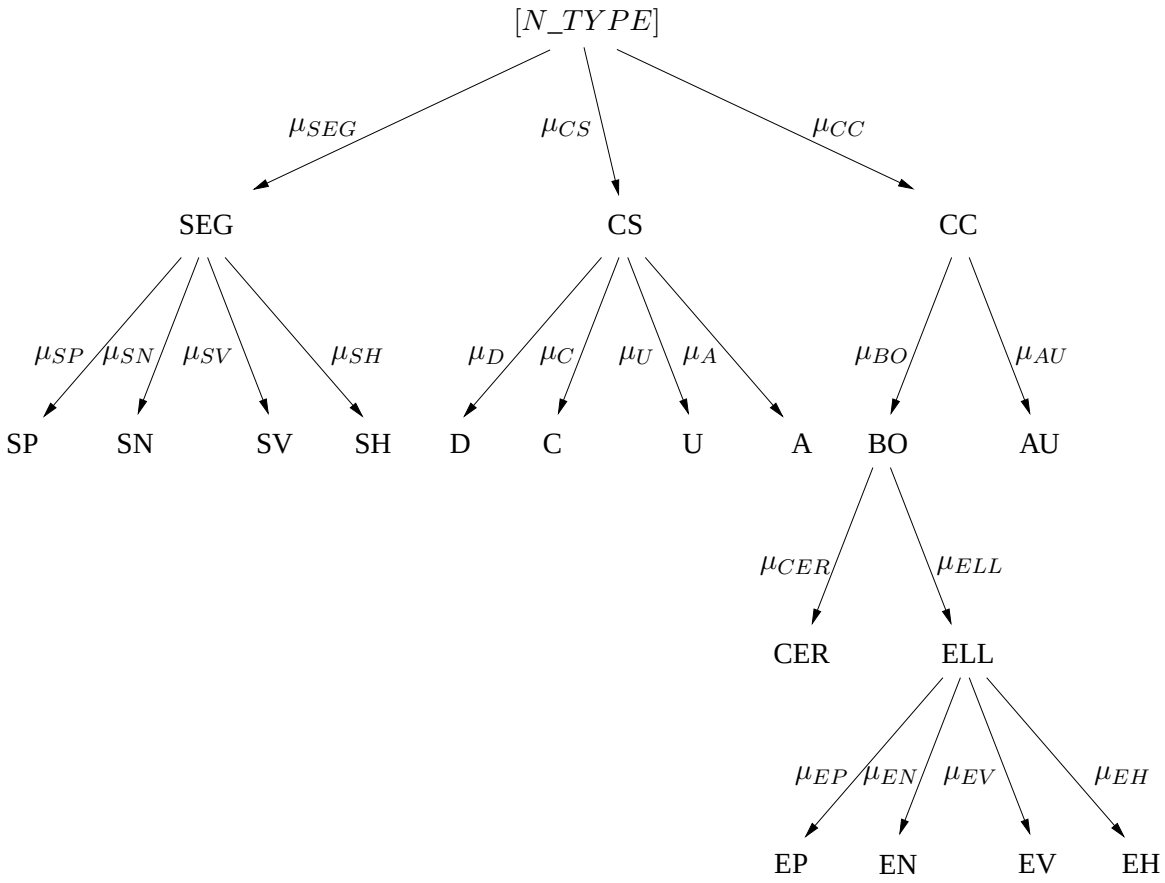


FIGURE 8.9 – Représentation de N_TYPE.

N_TYPE : Le type des primitives est décrit par un attribut hiérarchique de profondeur 4. Il contient six variables linguistiques réparties sur les différents niveaux et correspondant à une propriété particulière de la primitive spécifiée au niveau supérieur de l'attribut. La figure 8.9 donne la représentation de N_TYPE et la figure 8.10 page suivante explicite les propriétés des primitives associées aux valeurs des différentes variables.

La variable du premier niveau est associée à la courbure (« Arcness », Ar_{AB}) de la primitive qui relie les points A et B. La courbure se calcule numériquement de la façon suivante ([PDM86]) :

$$Ar_{AB} = \left(1 - \frac{AB}{\overset{\frown}{AB}}\right)^{\frac{1}{2}} \tag{8.9}$$

avec AB la distance euclidienne entre A et B et $\overset{\frown}{AB}$ la longueur de l'arc. La variable associée peut prendre trois valeurs {Segment, Courbe Simple, Courbe Complexe}. Les fonctions d'appartenance sont données à la table 8.11 page suivante et représentées sur la figure 8.11 page suivante.

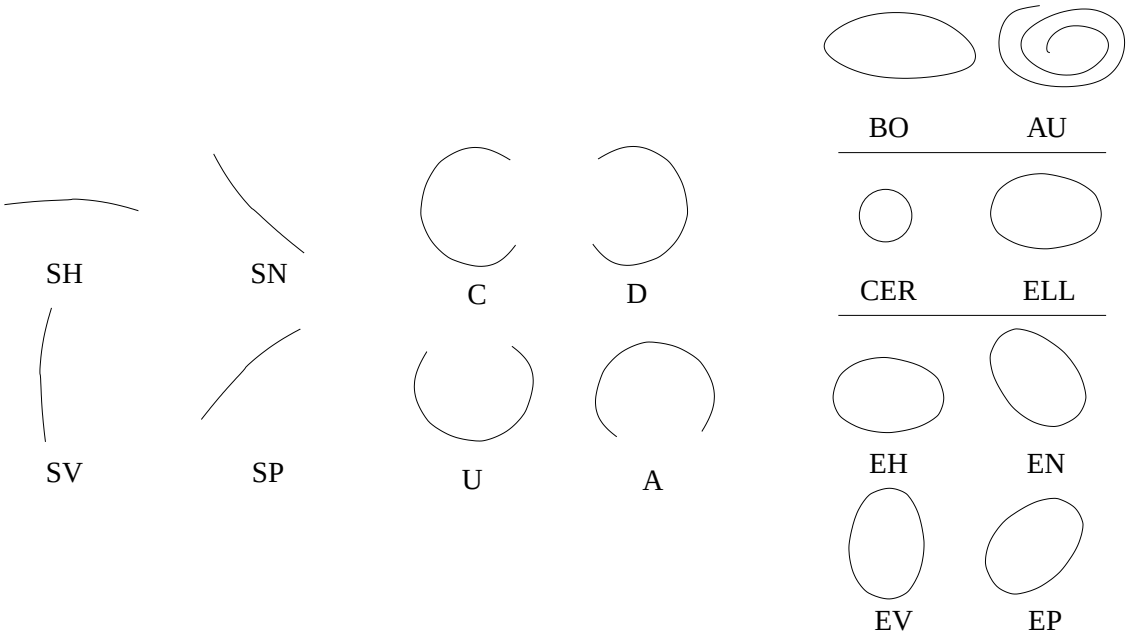


FIGURE 8.10 – Schéma des différents types de primitives décrites par N_TYPE.

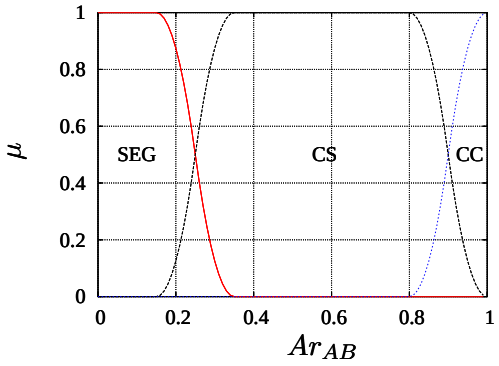
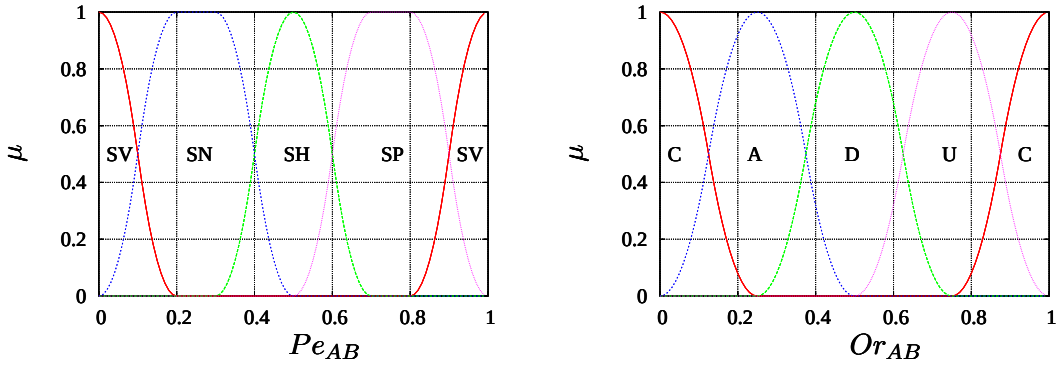


FIGURE 8.11 – Fonctions d'appartenance de la variable linguistique associée à la courbure Ar_{AB}.

Valeur	Symbole	Paramètres	Fonction
Segment	SEG	$a = 0,3\,;\, b = 0,4\,;\, c = 0,5$	$\mu_{SEG}(x) = 1 - S(x; a, b, c)$
Courbe Simple	CS	$a_1 = 0,3\,;\, b_1 = 0,4\,;\, c_1 = 0,5$ $a_2 = 0,7\,;\, b_2 = 0,8\,;\, c_2 = 0,9$	$\mu_{CS}(x) = \min\{S(x; a_1, b_1, c_1),$ $1 - S(x; a_2, b_2, c_2)\}$
Courbe Complexe	CC	$a = 0,7\,;\, b = 0,8\,;\, c = 0,9$	$\mu_{CC}(x) = 1 - S(x; a, b, c)$

TABLE 8.2 – Fonctions d'appartenance de la variable linguistique associée à la courbure Ar_{AB}.

Les trois variables du deuxième niveau de N_TYPE sont associées respectivement à la pente du segment (notée Pe_{AB}), à l'orientation de la courbe simple (notée Or_{AB}) et à ce que nous avons appelé le facteur de complexité de la courbe (notée Co_{AB}). La pente du segment $[AB]$ est notée Pe_{AB}, elle est normalisée sur $[0, 1]$ (0 correspondant à une pente de 0°et 1 à une pente de 180°par rapport à l'horizontale). Les fonctions d'appartenance de la variable associée à Pe_{AB} sont données à la table 8.3 page



(a) (b)

FIGURE 8.12 – Fonctions d'appartenance des variables linguistiques associées à : (a) la pente Pe_{AB}, (b) l'orientation de la courbe Or_{AB}.

suivante et représentées sur la figure 8.12 (a). Le support des pentes horizontales et verticales est volontairement plus étroit que celui des pentes positives et négatives afin de privilégier cette information lors de la reconnaissance. L'orientation d'une courbe simple est donnée par la pente orientée normalisée sur $[0, 1]$ (0 correspondant à une pente de 0°et 1 à une pente de 360°par rapport à l'horizontale) du segment $[GC]$ où G est le centre de gravité de la courbe et C le milieu du segment AB (figure 8.13 page suivante (a)). Les fonctions d'appartenance de la variable associée à Or_{AB} sont données à la table 8.4 page suivante et représentées sur la figure 8.12 (b). Les supports des quatre sous-ensembles flous ont, cette fois-ci, été choisis égaux puisqu'il n'y a aucune raison de privilégier une orientation particulière. Enfin la complexité des courbes fermées est calculée grâce au ratio de la longueur de l'ellipse de Fourier sur la longueur de la primitive (figure 8.13 page suivante (b)). L'ellipse de Fourier est la fréquence fondamentale du signal complexe formé par les abscisses et les ordonnées des points de la primitive parcourue de A à B. Elle s'obtient facilement par reconstruction (une transformée de Fourier inverse) de la primitive à partir des coefficients a_1 et a_{-1} définis au paragraphe 2.2.2.2 page 30. Les fonctions d'appartenance sont présentées à la table 8.5 page suivante et sur la figure 8.14 page 135. Les supports ont été déterminés empiriquement sur des exemples simples.

Valeur	Symbole	Paramètres	Fonction
Segment Vertical	SV	$a = 0\,;\, b = 0,2\,;\, c = 1$	$\mu_{SV}(x) = \max\{\Pi(x;a,b), \Pi(x;c,b)\}$
Segment Horizontal	SH	$a = 0,5\,;\, b = 0,2$	$\mu_{SV}(x) = \Pi(x;a,b)$
Segment Negatif	SN	$a_1 = 0\,;\, b_1 = 0,1\,;\, c_1 = 0,2$ $a_2 = 0,3\,;\, b_2 = 0,4\,;\, c_2 = 0,5$	$\mu_{CS}(x) = \min\{S(x;a_1,b_1,c_1),$ $1 - S(x;a_2,b_2,c_2)\}$
Segment Positif	SP	$a_1 = 0,5\,;\, b_1 = 0,6\,;\, c_1 = 0,7$ $a_2 = 0,8\,;\, b_2 = 0,9\,;\, c_2 = 1,0$	$\mu_{CS}(x) = \min\{S(x;a_1,b_1,c_1),$ $1 - S(x;a_2,b_2,c_2)\}$

TABLE 8.3 – Fonctions d'appartenance de la variable linguistique associée à la pente Pe_{AB}.

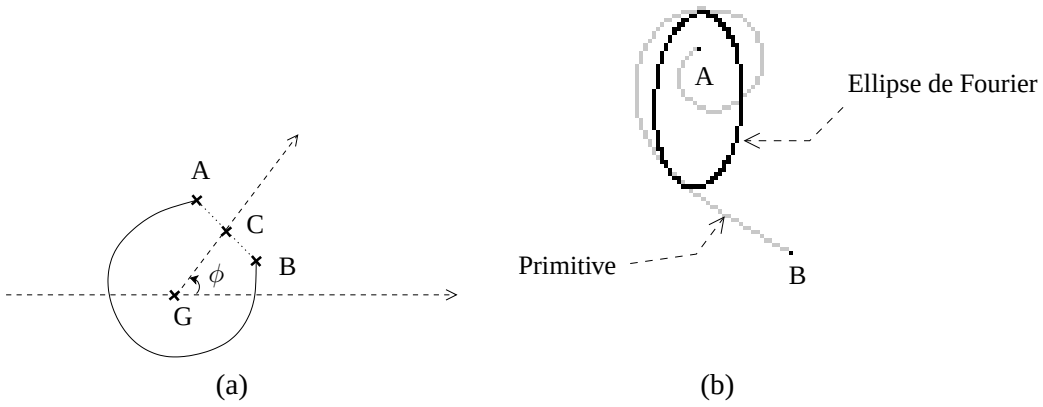


FIGURE 8.13 – (a) Calcul de l'orientation d'une courbe simple, $Or_{AB} = \frac{\phi}{2\pi}$. (b) Ellipse de Fourier pour une primitive complexe, $Co_{AB} = \frac{\text{Périmètre de l'Ellipse}}{\widehat{AB}}$.

Valeur	Symbole	Paramètres	Fonction
Courbe type C	C	$a = 0\,;\, b = 0,25\,;\, c = 1$	$\mu_C(x) = \max\{\Pi(x;a,b), \Pi(x;c,b)\}$
Courbe type A	A	$a = 0,25\,;\, b = 0,25$	$\mu_A(x) = \Pi(x;a,b)$
Courbe type D	D	$a = 0,5\,;\, b = 0,25$	$\mu_D(x) = \Pi(x;a,b)$
Courbe type U	U	$a = 0,75\,;\, b = 0,25$	$\mu_U(x) = \Pi(x;a,b)$

TABLE 8.4 – Fonctions d'appartenance de la variable linguistique associée à la complexité de la courbe Or_{AB}.

Valeur	Symbole	Paramètres	Fonction
Boucle	BO	$a = 0,8\,;\, b = 0,85\,;\, b = 0,95$	$\mu_{BO}(x) = S(x;a,b,c)$
Autres	AU	$a = 0,8\,;\, b = 0,85\,;\, b = 0,95$	$\mu_{AU}(x) = 1 - S(x;a,b,c)$

TABLE 8.5 – Fonctions d'appartenance de la variable linguistique associée à la complexité de la courbe Co_{AB}.

Le troisième niveau de N_TYPE distingue au sein des boucles, les cercles des ellipses. Nous utilisons « l'ellipsoïdité » (notée Ell_{AB}) de la primitive calculée comme étant le rapport de la longueur du petit axe sur celle du grand axe de l'ellipse de Fourier. Les fonctions d'appartenance sont données à la table 8.6 et sur la figure 8.15 page suivante. Les supports ont, là encore, été déterminés empiriquement sur des exemples simples.

Valeur	Symbole	Paramètres	Fonction
Cercle	CER	$a = 0,7\,;\, b = 0,8\,;\, b = 0,9$	$\mu_{CER}(x) = S(x;a,b,c)$
Ellipse	ELL	$a = 0,7\,;\, b = 0,8\,;\, b = 0,9$	$\mu_{ELL}(x) = 1 - S(x;a,b,c)$

TABLE 8.6 – Fonctions d'appartenance de la variable linguistique associée à l'ellipsoïdité $EllAB$.

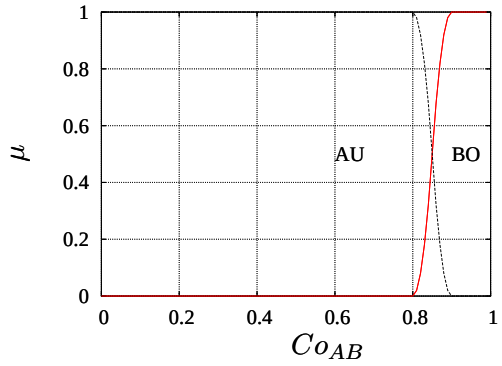


FIGURE 8.14 – Fonctions d'appartenance de la variable linguistique associée à la complexité de la courbe Co_{AB}.

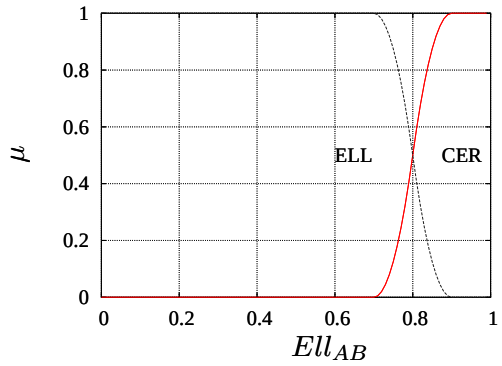


FIGURE 8.15 – Fonctions d'appartenance de la variable linguistique associée à l'ellipsoïdité Ell_{AB}.

Enfin le quatrième niveau de N_TYPE concerne l'orientation des ellipses. La variable linguistique est associée à la pente du grand axe de l'ellipse normalisée sur $[0, 1]$. La « fuzzification » est réalisée de la même façon que pour la pente des segments.

8.2.4.3 Attributs des arcs

Deux attributs de profondeur 1 sont associés aux arcs, A_PROX et A_POS. Le premier exprime la proximité floue et le second la position relative floue entre les nœuds qu'ils relient. Leurs représentations sont données sur la figure 8.16.

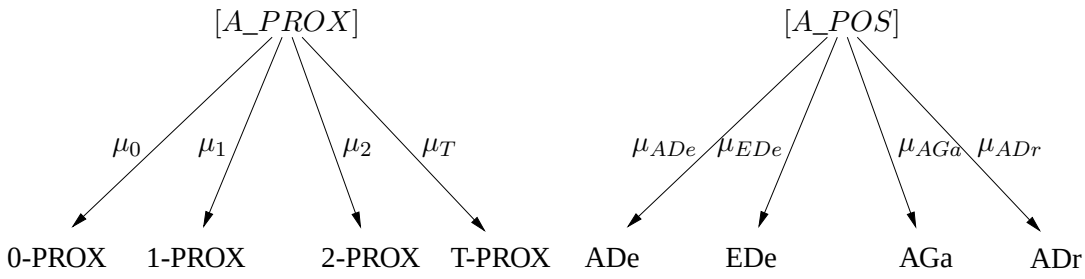


FIGURE 8.16 – Représentation de **A_PROX** et **A_POS**

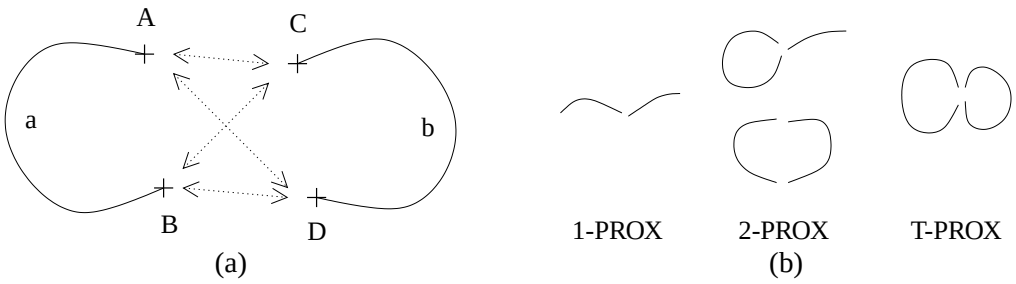


FIGURE 8.17 – (a) Distances utilisées dans le calcul de A_PROX entre deux primitives a et b (les distances sont représentées en pointillés). (b) Exemples de différentes connexités.

A_PROX : La variable linguistique associée à l'attribut de proximité peut prendre quatre valeurs appelées {0-PROX, 1-PROX, 2-PROX, T-PROX} qui correspondent à la proximité des quatre points de fin des primitives a et b que relie l'arc considéré (figure 8.17 (a)). Les quatre distances sont comparées au quart de la somme des longueurs des deux primitives. Si aucune de ces distances n'est petite, les primitives sont considérées 0-PROX, si une de ces distances est petite les primitives sont 1-PROX, si deux distances sont petites les primitives sont 2-PROX et si toutes les distances sont petites les primitives sont T-PROX. Pour le calcul des fonctions d'appartenance, les distances sont tout d'abord divisées par le quart de la somme des longueurs des primitives. Si la quantité obtenue dépasse 1, elle est mise à 1. Les quatre distances normalisées ainsi obtenues sont rangées dans l'ordre croissant et numérotées d_1, d_2, d_3 et d_4 avec $d_1 \leq d_2 \leq d_3 \leq d_4$. Ensuite les fonctions d'appartenance sont calculées comme indiqué dans la table 8.7. La figure 8.17 (b) donne quelques exemples de proximités. La description comporte une ambiguïté sur la 2-proximité mais qui n'a pas d'influence sur la reconnaissance puisqu'elle implique deux types de primitives complètement différentes. L'ambiguïté est levée par N_TYPE.

Valeur	Symbole	Fonction
Proximité nulle	0-PROX	$\mu_0 = d_1$
Proximité de type 1	1-PROX	$\mu_1 = \min(1 - d_1, d_2)$
Proximité de type 2	2-PROX	$\mu_2 = \min(1 - d_2, d_3)$
Proximité totale	T-PROX	$\mu_A = 1 - d_3$

TABLE 8.7 – Fonctions d'appartenance de la variable linguistique associée à l'attribut de proximité.

A_POS : Pour décrire les positions relatives de deux primitives, nous avons utilisé la définition proposée par BLOCH ([Blo99]). Celle-ci propose une méthode générale pour calculer les degrés d'appartenance d'une variable linguistique décrivant les positions relatives d'un objet flou par rapport à un autre objet flou de l'espace suivant une direction donnée. Cette définition est intéressante pour son invariance en translation, rotation et changement d'échelle. Dans notre cas, les objets sont dans un plan et non flous ce qui simplifie le calcul tout en conservant la validité de l'approche. Considérons deux primitives a et b de l'image de squelette et notons A les points de a et B les points de b. Notons $\beta(A, B)$ l'angle entre le vecteur $\vec{AB}$ et la direction $\vec{u_\alpha}$ où α représente l'angle de la direction considérée par rapport à l'horizontale. L'angle β se calcule sur $[0, \pi]$ par :

$$\begin{cases} \beta(A,B) = \arccos\left[\frac{\vec{AP}\cdot\vec{u_\alpha}}{\|\vec{AB}\|}\right] \text{ si } A \neq B \\ \beta(A,A) = 0 \end{cases} \tag{8.10}$$

Notons maintenant $\beta_{\min}(A)$ la valeur de $\beta(AB)$ obtenue pour le point B de b le plus proche de la droite passant par A dans la direction α :

$$\beta_{\min}(A) = \min_B(\beta(AB)) \tag{8.11}$$

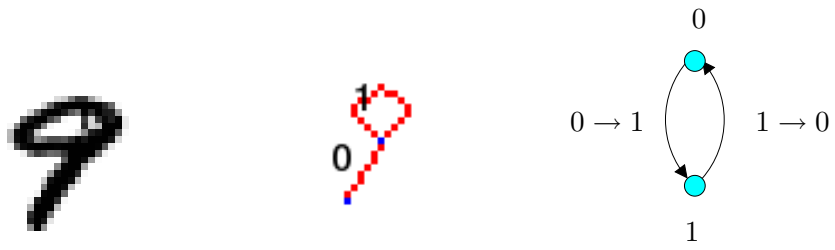


FIGURE 8.18 – Décomposition d'un caractère « 9 » issu de la base MNIST.

BLOCH définit alors le paysage flou $\mu_\alpha(b)(A)$ dans la direction α au point A comme étant une fonction décroissante de $\beta_{\min}(A)$ allant de $[0, \pi]$ dans $[0, 1]$ et propose d'utiliser la fonction linéaire suivante :

$$\mu_\alpha(b)(A) = \max\left(0, 1 - \frac{2\beta_{\min}(A)}{\pi}\right) \tag{8.12}$$

Le degré d'appartenance de l'arc (a, b) au sous-ensemble flou « a est dans la direction α de b » se calcule par combinaison des $\mu_\alpha(b)(A)$ pour tous les points A de a. L'évaluation optimiste utilisera une mesure de possibilité (la borne supérieure), l'évaluation pessimiste utilisera une mesure de nécessité (la borne inférieure). Nous avons choisi de conserver la moyenne qui est un bon compromis :

$$\mu_\alpha(b)(a) = \frac{1}{|a|} \sum_{A \in a} \mu_\alpha(b)(A) \tag{8.13}$$

En utilisant la moyenne, nous perdons la propriété de symétrie des degrés d'appartenance entre les arcs $a \rightarrow b$ et $b \rightarrow a$. Ceci n'est pas un problème puisque comme nous le verrons, les mesures de similarité sont basées sur le maximum des degrés d'appartenance entre arcs opposés.

Nous avons utilisé quatre valeurs pour la variable linguistique associée à l'attribut de position relative : { A Gauche de, A Droite de, Au Dessus de, En Dessous de }. La table 8.8 donne les fonctions d'appartenance pour chacun des sous-ensembles flous associés.

Valeur	Symbole	Angle	Fonction
A Gauche de	AGa	0	$\mu_{AGa} = \mu_0(b)(a)$
A Droite de	ADr	π	$\mu_{ADr} = \mu_\pi(b)(a)$
En Dessous de	EDe	$\frac{\pi}{2}$	$\mu_{EDe} = \mu_\pi(b)(a)$
Au Dessus de	ADe	$\frac{3\pi}{2}$	$\mu_{ADe} = \mu_{3\pi/2}(b)(a)$

TABLE 8.8 – Fonctions d'appartenance de la variable linguistique associée à l'attribut de position.

8.2.5 Exemple

Prenons un exemple sur un « 9 » manuscrit issu de la base MNIST (voir l'annexe A page 155). La figure 8.18 donne le résultat des deux étapes de décomposition du caractère. Les tables 8.9, 8.10, 8.11 et 8.12 page suivante, présentent les attributs hiérarchiques calculés pour les deux nœuds et les deux arcs du GAHF. Les résultats obtenus sont bien cohérents et intuitifs et ce malgré la non-symétrie des attributs de positions relatives.

Attribut	Profondeur	Variable Linguistique	Valeur	Degré d'appartenance
N_LONG	1	longueur relative	G	$0,12$
			TG	$0,87$
			TTG	$0,50$
N_TYPE	1	Courbure	SEG	1
	2	Pente	SV	$0,07$
	2	Pente	SP	$0,93$

TABLE 8.9 – Attributs du nœud 0

Attribut	Profondeur	Variable Linguistique	Valeur	Degré d'appartenance
N_LONG	1	longueur relative	TTG	$0,82$
			E	$0,59$
N_TYPE	1	Courbure	CC	1
	2	Complexité	BOU	1
	3	Ellipsoïdité	CER	1

TABLE 8.10 – Attributs du nœud 1

Attribut	Profondeur	Variable Linguistique	Valeur	Degré d'appartenance
N_PROX	1	Proximité	2-PROX	1
N_POS	1	Position	AGa	$0,28$
			ADr	$0,16$
			EDe	$0,88$

TABLE 8.11 – Attributs de l'arc $(0 \rightarrow 1)$

Attribut	Profondeur	Variable Linguistique	Valeur	Degré d'appartenance
N_PROX	1	Proximité	2-PROX	1
N_POS	1	Position	AGa	$0,18$
			ADr	$0,42$
			ADe	$0,99$

TABLE 8.12 – Attributs de l'arc $(1 \rightarrow 0)$

8.3 Adaptation du système de reconnaissance aux GAHF

L'utilisation des GAHF dans un système de reconnaissance basé sur l'appariement de structures graphiques passe par la définition d'une mesure de comparaison entre les éléments des graphes à apparier. Ensuite la maximisation d'une mesure de comparaison globale entre deux graphes va guider la décision.

8.3.1 Comparaison d'ensembles flous

BOUCHON-MEUNIER ET COLL. ([BMRB96]) proposent une définition générale des mesures de comparaison d'ensembles flous. Nous redonnons quelques définitions avant d'introduire leur notion de M-mesure de comparaison.

Définition 38 (Mesure floue) *Une mesure floue M est une relation*[3] *de $F(\Omega)$ dans $\mathbb{R}^+$ telle que pour tout A et B de $F(\Omega)$:*
- *$M(\emptyset) = 0$,*
- *si $B \subseteq A$, alors $M(B) \leq M(A)$.*

L'inclusion est prise ici au sens de Zadeh (définition 49 page 177). Si M est à valeurs dans $[0, 1]$, il s'agit d'une mesure d'ensembles flous. Un exemple de mesure floue est la cardinalité d'un sous-ensemble flou définie par :

$$M_c(A) = \sum_{x \in \Omega} \mu_A(x) \tag{8.14}$$

Définition 39 (Différence de sous-ensembles flous) *Une opération sur $F(\Omega)$ est une différence, notée $-$, si elle satisfait, pour tout A et B de $F(\Omega)$:*
- *si $A \subseteq B$, alors $A - B = \emptyset$.*
- *$B - A$ est monotone par rapport à B, c.-à-d. $B \subseteq B' \Rightarrow B - A \subseteq B' - A$.*

Un exemple de différence est donné par :

$$\mu_{A-B}(x) = \max(0, \mu_A(x) - \mu_B(x)) \tag{8.15}$$

Définition 40 (M-mesure de comparaison) *Une M-mesure de comparaison sur Ω est une fonction $S : F(\Omega) \times F(\Omega) \rightarrow [0, 1]$ telle que $S(A, B) = F_S(M(A \cap B), M(B - A), M(A - B))$ pour une fonction $F_S : F(\Omega) \times F(\Omega) \times F(\Omega) \rightarrow [0, 1]$ et une mesure floue M sur Ω.*

Alors qu'une mesure de comparaison classique repose sur une fonction définie directement sur l'intersection et les différences des sous-ensembles flous, cette définition utilise une mesure floue de ces grandeurs. Cette approche est intéressante car elle permet de passer d'une relation d'ordre à une autre sans avoir à redémontrer les propriétés des mesures utilisées. BOUCHON-MEUNIER ET COLL. définissent ensuite quelques M-mesures particulières dont les M-mesures de ressemblance :

Définition 41 (M-mesure de similitude) *Une M-mesure de similitude sur Ω est une M-mesure de comparaison telle que $F_S(u, v, w)$ soit non-décroissante en u et non-croissante en v et w.*

Définition 42 (M-mesure de ressemblance) *Une M-mesure de ressemblance sur Ω est une M-mesure de similitude S qui possède les propriétés de :*
- *symétrie, $S(A, B) = S(B, A)$*
- *et réflexivité $S(A, A) = 1$.*

A partir de cette définition, si nous considérons la grandeur suivante :

$$\alpha(A, B) = \frac{M_c(A \cap B)}{M_c(A \cup B)} \tag{8.16}$$

où M_c est la mesure de cardinalité (equation 8.14). En décomposant $M(A \cup B) = M(A \cap B) + M(A - B) + M(B - A)$ à l'aide de la différence définie à l'équation 8.15, il est facile de prouver que α est une M-mesure de ressemblance ([BMRB96]). En utilisant l'intersection et l'union floues au sens de Zadeh (le min et max) α s'exprime :

$$\alpha(A, B) = \frac{\sum_{x \in \Omega} (\mu_A(x) \wedge \mu_B(x))}{\sum_{x \in \Omega} (\mu_A(x) \vee \mu_B(x))} \tag{8.17}$$

Cette mesure de similarité est utilisée par MAN ET POON ([MP93]). L'avantage de passer par cette formulation est qu'elle permet d'adapter la similarité à l'application en changeant de mesure floue.

[3] voir Annexe D page 175

8.3.2 Similarités entre les éléments de deux GAHF

8.3.2.1 Comparaisons d'attributs flous

Nous avons vu au paragraphe précédent comment définir une mesure de similarité entre deux sous ensembles flous d'un univers Ω. Or la comparaison de deux graphes d'attributs flous au sens de CHAN ET CHEUNG passe par la comparaison de deux attributs flous, réalisations d'une variable linguistique. Prenons l'exemple des attributs flous caractérisant la taille d'un segment. Les attributs seront de la forme $\{(Petite, \mu^i_P), (Moyenne, \mu^i_M), (Grande, \mu^i_G)\}$ où $i = 1$ ou 2 indique si le degré d'appartenance caractérise le premier ou le second des attributs à comparer. Nous voyons que la problématique est légèrement différente d'une mesure de similarité entre deux sous-ensembles flous.

BOUCHON-MEUNIER ET COLL. ([BMRB96]) proposent une définition générale pour ce type de problème. Considérons maintenant que Ω est un ensemble d'attributs[4] décrits au moyens de valeurs linguistiques définies sur des ensembles flous $\Omega_1, \Omega_2, \ldots, \Omega_n$. Un attribut A est associé à n valeurs $A_1, A_2, \ldots, A_n$ définies respectivement sur $\Omega_1, \Omega_2, \ldots, \Omega_n$. Nous supposons qu'une mesure floue est définie sur chaque Ω_i, notée M^i ainsi qu'une M^i-mesure de similitude notée S^i. Considérons maintenant un autre attribut B associé à n valeurs $B_1, B_2, \ldots, B_n$, il est dit que :

- $A \subseteq B$ si et seulement si A_i est inclus dans B_i pour $i = 1, \ldots, n$,
- l'intersection de A et B est définie comme un attribut $A \cap B$ dont les valeurs sont les $A_i \cap B_i$ $i = 1, \ldots, n$,
- la différence entre A et B est définie comme un attribut $A - B$ dont les valeurs sont les $A_i - B_i$ $i = 1, \ldots, n$,
- Une mesure d'information $\mathcal{M}$ est une application définie sur Ω et à valeurs dans $\mathbb{R}^{+n}$ telle que pour tout A de Ω :

$$\mathcal{M}(A) = \left(M^1(A_1), M^2(A_2), \ldots, M^n(A_n)\right) \tag{8.18}$$

$\mathcal{M}$ est une généralisation des mesures floues sur Ω. Elle est monotone, si $A \subseteq B$ alors $M^i(A_i) \leq M^i(B_i)$ $i = 1, \ldots, n$, ce qui se note $\mathcal{M}(A) \leq \mathcal{M}(B)$. De la même manière il est possible de définir les $\mathcal{M}$-mesures de similitudes :

Proposition 4 *Pour toute t-norme T, la mesure définie pour toute paire d'attributs de Ω par :*

$$S(A, B) = T\left(S^1(A_1, B_1), \ldots, S^n(A_n, B_n)\right) \tag{8.19}$$

est une $\mathcal{M}$-mesure de similitude.

En prenant la t-norme de Zadeh (le min), la $\mathcal{M}$-mesure de similitude obtenue s'écrit :

$$S(A, B) = \bigvee_{i=1}^{n} (\mu_{A_i} \wedge \mu_{B_i}) \tag{8.20}$$

et nous retrouvons les mesures utilisées par CHAN ET CHEUNG ([CC92]) pour la comparaison des attributs flous.

[4] Dans l'article original les attributs sont des objets et les valeurs linguistiques des attributs, mais comme le terme d'attribut est déjà utilisé pour les graphes flous avec le même sens que les objets de BOUCHON-MEUNIER ET COLL., nous avons gardé le même formalisme par souci de clarté

L'utilisation du min étant très restrictive, MAN ET POON ([MP93]) proposent une mesure de similitude entre deux attributs flous qui s'exprime par :

$$\alpha_e(A,B) = \frac{\sum_{i=1}^{n}(\mu_{A_i} \wedge \mu_{B_i})}{\sum_{i=1}^{n}(\mu_{A_i} \vee \mu_{B_i})} \tag{8.21}$$

8.3.2.2 Comparaison d'éléments hiérarchiques flous

Il est possible à partir d'un arbre linguistique flou et pour une profondeur donnée d de construire un ensemble de valeurs linguistiques en effectuant un produit cartésien[5] sur chaque branche. Ainsi, en reprenant l'attribut hiérarchique N_TYPE (figure 8.9 page 131), pour une profondeur $d = 2$, l'ensemble de valeurs linguistiques extraites sera $\{(SEG|SP, \mu_{SEG} \wedge \mu_{SP}),\ (SEG|SN, \mu_{SEG} \wedge \mu_{SN}),\ (SEG|SV, \mu_{SEG} \wedge \mu_{SV}), (SEG|SH, \mu_{SEG} \wedge \mu_{SH}), (CS|D, \mu_{CS} \wedge \mu_D), (CS|C, \mu_{CS} \wedge \mu_C), (CS|U, \mu_{CS} \wedge \mu_U), (CS|A, \mu_{CS} \wedge \mu_A), (CC|BO, \mu_{CC} \wedge \mu_{BO}), (CC|AU, \mu_{CC} \wedge \mu_{AU})\}$. Ainsi développés, les attributs hiérarchiques sont ramenés à des attributs flous dont les similitudes peuvent être calculées en utilisant α_e (équation 8.21).

La similitude entre deux nœuds hiérarchiques $\hat{n_1}$ et $\hat{n_2}$ à une profondeur d que nous avons utilisée se définit par :

$$\alpha_1(\hat{n_1}, \hat{n_2}) = \frac{\sum_{s}(\mu_{\hat{n_{1_s}}} \wedge \mu_{\hat{n_{2_s}}})}{\sum_{s}(\mu_{\hat{n_{1_s}}} \vee \mu_{\hat{n_{2_s}}})} \tag{8.22}$$

où les $\hat{n_{1s}}$ représentent toutes les valeurs linguistiques de profondeur d construites pour tous les attributs hiérarchiques du nœud $\hat{n_1}$. L'indice s est la chaîne de caractères qui indique les valeurs comparées ($SEG|SP$, $SEG|SN$, ...). De la même façon, la similitude que nous avons utilisée entre deux arcs hiérarchiques $\hat{a_1}$ et $\hat{a_2}$ à une profondeur d se définit par :

$$\alpha_2(\hat{a_1}, \hat{a_2}) = \frac{\sum_{s}(\mu_{\hat{a_{1_s}}} \wedge \mu_{\hat{a_{2_s}}})}{\sum_{s}(\mu_{\hat{a_{1_s}}} \vee \mu_{\hat{a_{2_s}}})} \tag{8.23}$$

Comme nous l'avons vu au paragraphe 8.2.4.3 page 136, l'attribut de position relative n'est pas symétrique. C'est pourquoi il est plus judicieux de comparer directement les couples d'arcs hiérarchiques de sens opposés. Un arc hiérarchique $\hat{a_1}$ et son arc opposé $\hat{a_1}^o$ sont regroupés dans un couple $\hat{P_{a_1}} = (\hat{a_1}, \hat{a_1}^o)$. Pour une profondeur d, la similitude de deux couples d'arcs hiérarchiques $\hat{P_{a_1}}$ et $\hat{P_{a_2}}$ se calcule simplement par :

$$\alpha'_2(\hat{P_{a_1}}, \hat{P_{a_2}}) = \max\{\alpha_2(\hat{a_1}, \hat{a_2}), \alpha_2(\hat{a_1}, \hat{a_2}^o), \alpha_2(\hat{a_1}^o, \hat{a_2}), \alpha_2(\hat{a_1}^o, \hat{a_2}^o)\} \tag{8.24}$$

8.3.3 Comparaison de GAHF

A partir des mesures de similitude entre éléments hiérarchiques, il est maintenant possible de réaliser un appariement en utilisant un des algorithmes présentés au paragraphe 6.3.3 page 96. Il suffit pour cela d'utiliser la similitude comme poids du graphe d'association généralisé. Notons h l'isomorphisme approximatif de sous-graphe obtenu entre deux GAHF $\hat{Ga_1} = (\hat{V_1}, \hat{E_1})$ et $\hat{Ga_2} = (\hat{V_2}, \hat{E_2})$ par un tel appariement. Une mesure de comparaison associée à h mesure la qualité de l'appariement entre $\hat{Ga_1}$

[5] voir annexe D page 175

et $\hat{Ga}_2$. Elle est appelée degré d'appariement par CHAN ET CHEUNG ([CC92]) qui la définissent de la façon suivante :

$$\gamma_{CC}(\hat{Ga}_1, \hat{Ga}_2) = \bigwedge_{i=1}^{|\hat{V_1}^h|} \alpha_1(\hat{n_i}, h(\hat{n_i})) \times \bigwedge_{i=1}^{|\hat{E_1}^h|} \alpha'_2(\hat{P}_{a_i}, \hat{P}_{h(a_i)}) \tag{8.25}$$

où $\hat{V_1}^h$ et $\hat{E_1}^h$ représentent les éléments de $\hat{Ga}_1$ qui ont une image par h. Avec l'utilisation des algorithmes de relaxation floue ou d'assignement gradué, tous les nœuds du graphe de plus petite cardinalité auront une image par h et $|\hat{V_1}^h| = \min\{|\hat{V_1}|, |\hat{V_2}|\}$. Le i^e nœud hiérarchique de $\hat{V_1}^h$ est désigné par $\hat{n_i}$ et la i^e paire d'arcs regroupant les arcs d'orientation opposée de $\hat{E_1}^h$ est notée $\hat{P}_{a_i}$.

La mesure de CHAN ET CHEUNG, que nous noterons γ_{CC} est très contraignante puisqu'elle est nulle dès que deux éléments ne sont pas similaires. Nous proposons d'utiliser un opérateur de moyenne moins contraignant pour définir une fonction objectif que nous noterons γ_m :

$$\gamma_m(\hat{Ga}_1, \hat{Ga}_2) = \frac{1}{|\hat{V_1}^h| \times |\hat{E_1}^h|} \sum_{i=1}^{|\hat{V_1}^h|} \alpha_1(\hat{n_i}, h(\hat{n_i})) \times \sum_{i=1}^{|\hat{E_1}^h|} \alpha'_2(\hat{P}_{a_i}, \hat{P}_{h(a_i)}) \tag{8.26}$$

Une telle mesure de comparaison ne tient pas compte du nombre d'éléments non appariés. La version pondérée de γ_m notée γ_{mp} permet d'en tenir compte :

$$\gamma_{mp}(\hat{Ga}_1, \hat{Ga}_2) = \frac{(|\hat{V_1}^h| - \Delta_v)(|\hat{E_1}^h| - \Delta_e)}{|\hat{V_1}^h|^2 \times |\hat{E_1}^h|^2} \sum_{i=1}^{|\hat{V_1}^h|} \alpha_1(\hat{n_i}, h(\hat{n_i})) \times \sum_{i=1}^{|\hat{E_1}^h|} \alpha'_2(\hat{P}_{a_i}, \hat{P}_{h(a_i)}) \tag{8.27}$$

avec :

$$\Delta_v = \begin{cases} |\hat{V}_{max}| - |\hat{V_1}^h| \text{ si } |\hat{V}_{max}| - |\hat{V_1}^h| \leq |\hat{V_1}^h| \\ |\hat{V_1}^h| \text{ sinon} \end{cases} \tag{8.28}$$

où $|\hat{V}_{max}| = \max\{|\hat{V_1}|, |\hat{V_2}|\}$, et :

$$\Delta_e = \begin{cases} |\hat{E}_{max}| - |\hat{E_1}^h| \text{ si } |\hat{E}_{max}| - |\hat{E_1}^h| \leq |\hat{E_1}^h| \\ |\hat{E_1}^h| \text{ sinon} \end{cases} \tag{8.29}$$

où $|\hat{E}_{max}| = \max\{|\hat{E_1}|, |\hat{E_2}|\}$.

8.3.4 Schéma de reconnaissance

Le système mis au point pour la reconnaissance à base de GAHF se décompose en trois étapes (figure 8.19 page suivante). En premier lieu, le caractère inconnu est squelettisé, décomposé en primitives et structuré sous forme de GAHF. Dans un deuxième temps le GAHF inconnu est apparié à chaque modèle de classe et une mesure de comparaison est calculée pour chaque appariement. Enfin la décision est prise en comparant les différentes mesures de comparaison, le caractère est étiqueté comme appartenant à la classe donnant la mesure de comparaison maximum.

Le prochain chapitre exploite ce système de reconnaissance sur les bases de validation.

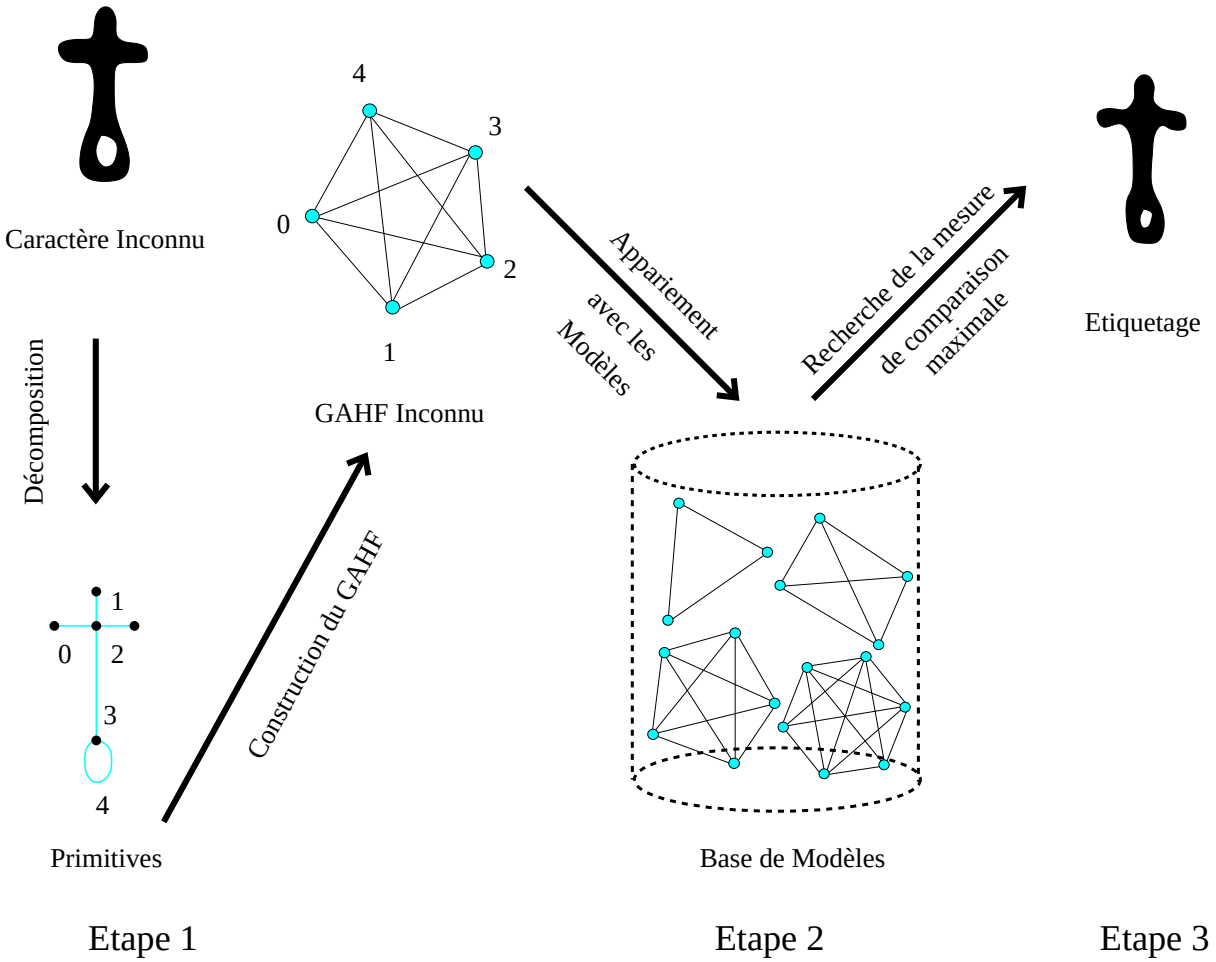


FIGURE 8.19 – Schéma de reconnaissance à base de GAHF en trois étapes.

CHAPITRE 9

RÉSULTATS ET DISCUSSION

Sommaire

Nous présentons dans ce chapitre quelques résultats obtenus sur les différentes bases avec une modélisation à base de GAHF. Les résultats sont donnés avec les deux algorithmes d'appariement (relaxation floue et assignement gradué) avec les mêmes paramètres que pour les graphes aléatoires soit :

- pour l'algorithme de GOLD ET COLL. par assignement gradué,
  - le facteur de pondération du poids des nœuds dans la fonction d'énergie, $\alpha = 0,5$,
  - la valeur initiale du paramètre de contrôle, $\beta_0 = 0,5$,
  - la valeur maximale du paramètre de contrôle, $\beta_f = 10$,
  - le taux de croissance du paramètre de contrôle, $\beta_r = 1,075$,
  - le critère de convergence pour chaque valeur de β, $\delta_1 = 0,05$,
  - le maximum d'itérations autorisées pour chaque valeur de β, $I_0 = 10$,
  - le critère de stabilisation sur la normalisation de M_{ai}, $\delta_0 = 0,5$,
  - le maximum d'itérations autorisées pour la stabilisation de M_{ai}, $I_1 = 30$,
- et pour l'algorithme de RANGARAJAN ET COLL. par relaxation floue,
  - le facteur d'attachement aux données initiales, $\alpha = 0,7$,
  - le seuil des poids du graphe d'association après relaxation, $Tlim = 0,5$,
  - le critère de convergence pour la relaxation, $\delta = 0,01$.

9.1 Résultats

Pour chaque algorithme d'appariement les résultats sont donnés avec les quatre niveaux de profondeur des arbres linguistiques. Les trois mesures de comparaison pour les GAHF ont été testées, γ_{CC} (équation 8.25 page 142), γ_m (équation 8.26 page 142) et γ_{mp} (équation 8.27 page 142).

9.1.1 Base GrCor

La table 9.1 donne les taux d'erreurs obtenus en validation croisée sur quatre groupes de 1 000 caractères pour la base GrCor. Rappelons qu'il n'y a pas d'apprentissage et que la validation croisée se résume donc à quatre séries de tests sur 1 000 individus tirés aléatoirement dans la base de laquelle ont bien sûr été retirés les modèles.

Algorithmes d'Appariement	GOLD ET COLL.				RANGARAJAN ET COLL.			
Profondeur	1	2	3	4	1	2	3	4
γ_m								
Erreur Moyenne	65,5%	60,0%	62,2%	62,5%	68,2%	65,9%	68,7%	69,5%
Ecart Type	2,3%	2,4%	2,4%	3,9%	3,9%	3,6%	2,4%	2,3%
γ_{mp}								
Erreur Moyenne	75,1%	71,0%	70,2%	70,2%	72,6%	70,9%	71,4%	71,7%
Ecart Type	2,9%	2,9%	2,5%	2,4%	2,0%	1,9%	2,6%	2,4%
γ_{CC}								
Erreur Moyenne	72,7%	72,0%	72,4%	72,2%	75,9%	74,9%	75,5%	75,2%
Ecart Type	1,9%	2,9%	3,3%	3,3%	1,5%	2,4%	2,9%	2,9%

TABLE 9.1 – Classification par GAHF avec la base GrCor

Les résultats sont globalement assez médiocres mais l'objectif est ici de regarder l'intérêt de l'approche en tant que méthode générique. Or ce qui est frappant c'est qu'on obtient, avec l'algorithme de GOLD ET COLL. et la mesure γ_m, des résultats comparables à ceux obtenus avec les graphes aléatoires d'arêtes sur la même base (table 7.7 page 112).

Sur cette base la mesure de comparaison la plus intéressante est la mesure par moyenne sans pondération par rapport aux éléments non appariés, γ_m. Comme les structures sont simples l'appariement suffit à les distinguer, la pondération va alors pénaliser les structures légèrement bruitées ou ayant un point d'inflexion supplémentaire. Comme prévu la mesure de CHAN ET CHEUNG, γ_{CC}, est trop stricte et conduit à des résultats décevants. Concernant la profondeur, il semble qu'au delà du niveau 2 l'information ajoutée n'aide plus à la reconnaissance mais commence même à la détériorer.

9.1.2 Base MNIST

Nous présentons dans la table 9.2 les résultats de classification obtenus sur la base MNIST avec quatre groupes de 1 000 individus.

Algorithmes d'Appariement	GOLD ET COLL.				RANGARAJAN ET COLL.			
Profondeur	1	2	3	4	1	2	3	4
γ_m								
Erreur Moyenne	58,3%	56,7%	57,4%	60,2%	59,3%	61,0%	60,9%	63,3%
Ecart Type	1,2%	0,9%	1,0%	0,3%	1,3%	1,6%	1,6%	1,6%
γ_{mp}								
Erreur Moyenne	59.3%	57,9%	57,5%	56,9%	60,1%	57,9%	58,3%	57,5%
Ecart Type	1,3%	0,9%	0,9%	0,9%	1,6%	1,7%	1,2%	1,2%
γ_{CC}								
Erreur Moyenne	57,3%	56,5%	56,3%	56,5%	58,5%	58,5%	58,3%	58,3%
Ecart Type	1,4%	1,1%	1,1%	1,1%	0,7%	0,9%	1,0%	1,0%

TABLE 9.2 – Classification par GAHF avec la base MNIST

Les résultats sont tous assez homogènes autour de 58% d'erreurs. Pour cette base il semble que la profondeur, la mesure de comparaison ou encore l'algorithme d'appariement ne soient pas discriminants. Comme les structures sont assez simples le facteur limitant vient du bruit des structures. La table 9.3 donne la matrice de confusion obtenue avec γ_m, une profondeur de 2 et l'algorithme de GOLD ET COLL..

Classes	0	1	2	3	4	5	6	7	8	9	nombre d'éléments
0	**88**	0	0	0	0	3	0	2	5	1	410
1	0	**75**	0	0	1	0	0	0	1	21	428
2	2	0	**43**	9	3	11	5	19	4	4	394
3	4	0	26	**20**	1	4	2	34	5	4	420
4	1	10	3	4	**64**	8	1	3	2	4	388
5	2	1	21	14	2	**34**	8	10	7	2	382
6	5	16	9	4	11	15	**18**	3	8	12	390
7	1	0	9	22	4	8	1	**41**	8	5	404
8	10	12	2	3	4	6	10	6	**27**	19	373
9	28	20	2	3	3	2	1	13	6	**20**	411
Pertinence	0,62	0,56	0,37	0,25	0,69	0,37	0,39	0,31	0,37	0,22	

TABLE 9.3 – Matrice de confusion (en pourcentage de bonnes reconnaisssances) pour la classification par GAHF (γ_m, et profondeur de 2) avec la base MNIST et l'algorithme de GOLD ET COLL.. Les colonnes correspondent aux valeurs reconnues, les lignes aux valeurs vraies.

Là encore les faibles pertinences des classes montrent bien que le bruit des structures est le vrai facteur limitant rendant difficile l'interprétation des résultats. Les structures simples des chiffres '0' et '1' obtiennent des scores importants avec des pertinences raisonnables. Le chiffre '4' aussi semble avoir une structure assez discriminante. Notons enfin que comme pour la base GrCor, les scores sont comparables à ceux obtenus avec les graphes aléatoires d'arêtes.

9.1.3 Base GrAnc

Les résultats de classification pour la base GrAnc avec le système par GAHF sont présentés dans la table 9.4. Ils ont été obtenus à partir de quatre groupes de 200 individus.

Algorithmes d'Appariement	GOLD ET COLL.				RANGARAJAN ET COLL.			
Profondeur	1	2	3	4	1	2	3	4
γ_m								
Erreur Moyenne	69,7%	66,9%	67,1%	66,0%	72,2%	69,7%	70,1%	68,7%
Ecart Type	2,3%	3,5%	3,8%	4,6%	5,0%	4,2%	4,0%	4,5%
γ_{mp}								
Erreur Moyenne	84,5%	83,1%	83,4%	82,2%	84,2%	83,2%	83,6%	82,5%
Ecart Type	3,7%	4,2%	4,2%	4,2%	3,1%	3,8%	4,0%	4,1%
γ_{CC}								
Erreur Moyenne	87,7%	84,2%	84,4%	83,1%	86,2%	85,9%	86,2%	84,9%
Ecart Type	3,1%	3,3%	3,5%	3,2%	2,3%	2,2%	2,4%	1,8%

TABLE 9.4 – Classification par GAHF avec la base GrAnc

Les scores sont comparables à ceux obtenus avec la base GrCor. Les mesures γ_{mp} et γ_{CC} donnent des résultats nettement moins bons que la mesure γ_m. Au delà d'une profondeur de 2 l'information ajoutée n'apporte rien. Enfin les performances sont, là aussi, du même ordre que ceux obtenus avec les graphes aléatoires d'arêtes.

9.1.4 Base IFN/ENIT

Pour la base IFN/ENIT, les structures étant relativement bruitées, les appariements avec γ_{mp} et γ_{CC} donnent des taux d'erreurs au delà de 98% et nous ne les avons pas présentés ici. Par contre les résultats obtenus avec la mesure γ_m sont assez intéressants. Ils sont présentés dans la table 9.5. Les graphes pouvant être de taille relativement importante, les temps de calculs sont assez conséquents, c'est pourquoi nous n'avons utilisé que 400 caractères pour les tests et n'avons pas calculé d'écart type. Nous donnons par contre les pourcentages d'erreurs après rappels (le i^{e} rappel correspond au pourcentage d'erreur par rapport aux $i+1$ premiers résultats retournés).

Algorithmes d'Appariement	GOLD ET COLL.				RANGARAJAN ET COLL.			
Profondeur	1	2	3	4	1	2	3	4
Erreur initiale	75%	69%	71%	77%	91%	84%	78%	86%
Erreur après 1 rappel	65%	62%	58%	70%	83%	78%	67%	75%
Erreur après 2 rappels	59%	59%	54%	63%	75%	72%	58%	73%
Erreur après 3 rappels	58%	56%	48%	60%	70%	70%	55%	71%
Erreur après 4 rappels	55%	54%	45%	58%	68%	66%	51%	67%
Erreur après 5 rappels	53%	50%	44%	56%	66%	64%	50%	65%
Erreur après 6 rappels	50%	49%	43%	55%	60%	61%	49%	63%
Erreur après 7 rappels	49%	48%	40%	53%	58%	58%	45%	60%
Erreur après 8 rappels	48%	47%	38%	52%	56%	53%	42%	59%
Erreur après 9 rappels	45%	46%	36%	51%	53%	51%	42%	59%

TABLE 9.5 – Classification par GAHF avec la base IFN/ENIT

Tout d'abord, ces résultats confirment que l'algorithme de GOLD ET COLL. se comporte mieux que celui de RANGARAJAN ET COLL. et ce même sur des caractères complexes comme des mots arabes. Ensuite nous voyons que la profondeur des attributs a une importance et qu'une description trop complète peut nuire à la reconnaissance. Enfin, l'intérêt de ces résultats est de montrer que notre système, sans apprentissage donne des résultats certes médiocres mais qui ont un sens. A titre de comparaison, et toutes proportions gardées, certains systèmes dédiés avec apprentissage testés sur la base entière (c'est-à-dire avec 937 classes et non seulement 78 comme nous) donnent des résultats du même ordre voire moins bons ([MPEA05]).

9.1.5 Base HrMan

Voici quelques résultats d'indexation obtenus avec la base de hiéroglyphes manuscrits. Rappelons que les caractères inconnus sont issus d'un manuscrit différent de celui duquel sont extraits les caractères de la base. L'indexation a été effectuée avec l'algorithme de GOLD ET COLL. pour les mesures de comparaison γ_m et γ_{mp} (la mesure γ_{CC} n'étant pas pertinente pour cette base) et en utilisant simultanément une profondeur de 2 et de 4. Les quatre premières images retournées par le système sont données pour chaque image inconnue dans la table 9.6 page 150.

Les résultats sont intéressants sur deux aspects. D'abord ils sont globalement cohérents et distinguent des familles de structures plutôt intuitives. Ensuite ils sont assez représentatifs de la différence qu'introduisent les deux mesures de comparaison. La première, γ_m, s'intéresse uniquement à la similarité entre les éléments ce qui lui donne un avantage sur le deuxième et le douzième caractères par exemple. Par contre l'introduction des pondérations sur le nombre d'éléments appariés par la mesure γ_{mp} lui permet d'obtenir globalement un meilleur retour.

Concernant la profondeur de la description, elle n'a pas énormément d'influence sur ces caractères excepté sur le neuvième caractère où elle permet d'éliminer la deuxième réponse obtenue avec γ_m à la

profondeur 2.

9.2 Discussion

Plusieurs choses se dégagent de cette série de résultats. D'abord, il est évident que ce système sans apprentissage ne peut pas rivaliser avec des systèmes de classification dédiés sur des bases de caractères. Rappelons en plus que les paramètres d'appariement n'ont pas été optimisés et qu'il conviendrait de réaliser un plan d'expérience complet pour voir quelles peuvent être les meilleures performances de cette chaîne de traitement sur chacune des bases.

L'objectif principal de ces tests était de valider la généricité du traitement. Nous voyons que pour toutes les bases les résultats ont un sens et ce même sans connaissance *a priori*. De plus nous avons montré que l'adaptation à la base peut se faire par le choix d'une mesure de comparaison appropriée. Or dans ce domaine les possibilités sont vastes (en utilisant les $\mathcal{M}$-mesures de similitude de BOUCHON-MEUNIER ET COLL. ([BMRB96]) par exemple).

Pour ce qui est de l'intérêt de l'approche hiérarchique permettant de décrire plus ou moins en profondeur le type des primitives. Il y a peu de cas dans les bases traitées où une profondeur supérieure à 2 apporte une information utile au système. Malgré tout nous avons vu qu'une description plus profonde pouvait soit dégrader les performances soit, dans le cas du neuvième caractère égyptien, lever une ambiguïté. La hiérarchie peut participer de la généricité du système en lui permettant de s'adapter à l'écriture traitée.

Concernant les limitations de l'approche, elles sont encore nombreuses. Le bruit des structures gène considérablement l'appariement. Néanmoins si nous comparons les résultats obtenus avec la base GrCor et de la base MNIST, il semble que la multiplication des scripteurs ne soit pas un facteur limitant. C'est un point encourageant qui pousse à travailler encore sur l'extraction des primitives et le choix des modèles. Un autre facteur limitant est le temps de calcul dont nous n'avons pas parlé puisqu'il n'était pas au cœur de notre travail. Pour l'instant les tests sont relativement lents et ce pour deux raisons. D'une part l'appariement reste coûteux en temps et en mémoire et d'autre part l'implémentation du calcul des similarités entre éléments demande encore à être optimisée.

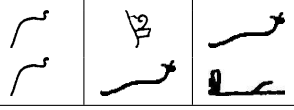
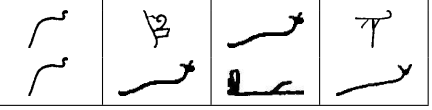
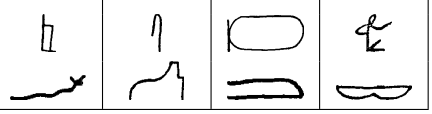
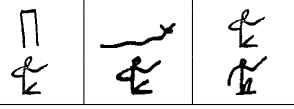
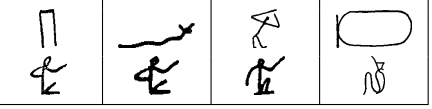
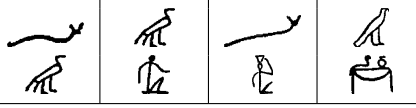
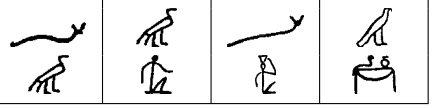
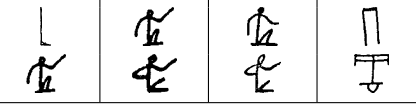
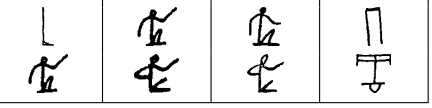
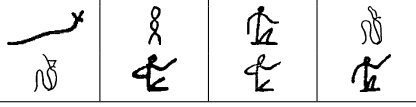
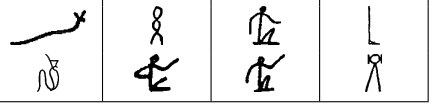
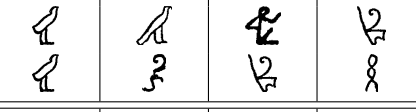
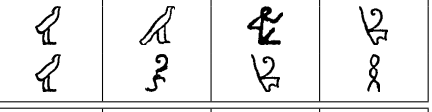
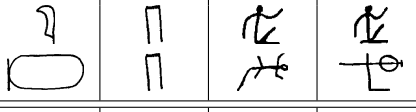
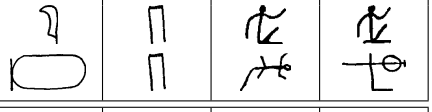
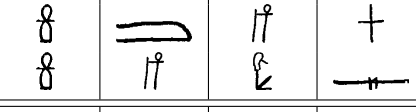
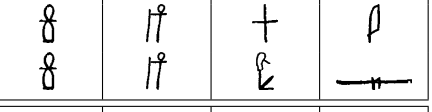
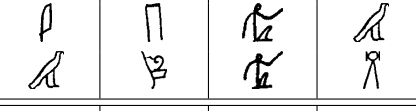
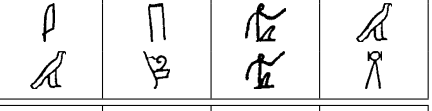
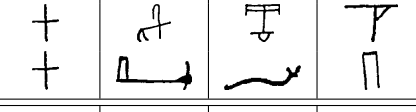
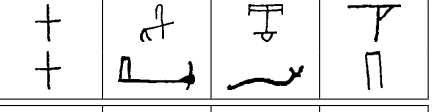
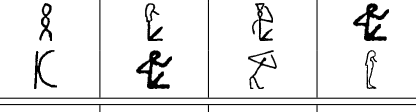
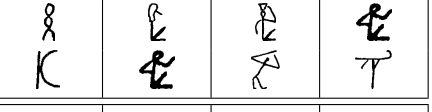
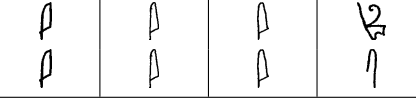
Index	Similarité	Inconnu	Profondeur 2				Profondeur 4			
1	γ_m γ_{mp}									
2	γ_m γ_{mp}									
3	γ_m γ_{mp}									
4	γ_m γ_{mp}									
5	γ_m γ_{mp}									
6	γ_m γ_{mp}									
7	γ_m γ_{mp}									
8	γ_m γ_{mp}									
9	γ_m γ_{mp}									
10	γ_m γ_{mp}									
11	γ_m γ_{mp}									
12	γ_m γ_{mp}									
13	γ_m γ_{mp}									

TABLE 9.6 – Résultats d'indexation avec la base HrMan

CHAPITRE 10

SYNTHÈSE, CONCLUSION GÉNÉRALE ET PERSPECTIVES

Sommaire

Ce manuscrit a exposé les travaux réalisés au cours de cette thèse de doctorat. Ils ont tourné autour de l'apport des structures graphiques en reconnaissance de caractères. Deux problématiques ont été abordées, la réalisation d'une chaîne de reconnaissance probabiliste avec apprentissage d'une part, et, d'autre part, la réalisation d'une chaîne de reconnaissance floue générique et sans apprentissage.

10.1 Synthèse

Nous avons commencé par replacer nos travaux dans le contexte de la reconnaissance de caractères qui est, comme nous l'avons vu en introduction de ce mémoire, une question encore ouverte sur différents types d'écritures manuscrites. Aujourd'hui, les problèmes à résoudre sont la reconnaissance d'écrits manuscrits multi et omni-scripteurs pour les écritures à vocabulaire réduit et étendu dans un cadre non-contraint. La problématique ultime est d'arriver à concevoir un système générique capable de s'adapter à tout type d'écriture. Nous avons, pour valider nos propositions, choisi de travailler sur cinq bases qui couvrent à peu près l'ensemble de ce spectre.

Afin de pouvoir comparer nos méthodes, au moins sur les bases de caractères à vocabulaire réduit, nous avons, dans la première partie, redonné quelques éléments classiques de reconnaissance statistique de formes et les avons appliqués sur nos bases.

Nous avons commencé, dans le deuxième chapitre, par exposer différents descripteurs utilisés dans ce contexte ainsi que quelques mesures de distance. Les descripteurs statistiques sont de deux types suivant qu'ils s'appliquent à décrire la forme ou son contour. Les distances, quant à elles, se déterminent par rapport au type des éléments qu'elles comparent.

Les classifieurs statistiques dits probabilistes, exposés dans le troisième chapitre, s'attachent à estimer les densités de probabilités liées aux classes de la base d'apprentissage. Suivant que l'approche utilise un modèle pour l'estimation ou qu'elle n'en utilise pas, les classifieurs sont dits paramétrés ou non. Nous rappelons le principe de fonctionnement des méthodes paramétrées en utilisant un modèle

gaussien et redonnons ensuite les définitions de trois méthodes non paramétrées classiques. Enfin nous réexpliquons brièvement le principe de décision bayésienne avec rejets qui nous a servi à réaliser nos tests.

Le quatrième et dernier chapitre de cette première partie, donne des résultats de reconnaissance sur les trois bases de caractères à vocabulaire réduit. Les conclusions de ces tests sont de deux ordres. Dans un premier temps, d'une manière générale, ils nous ont permis de nous rendre compte que les approches statistiques sont relativement efficaces et que les descripteurs de Zernike sont un excellent moyen de décrire ce genre de formes. Combinés avec les moments de Fourier complexes, ils donnent des résultats de reconnaissance relativement bons même avec une réduction des données par analyse en composantes principales. Enfin dans un deuxième temps, ces résultats nous ont montré que les scores de reconnaissance statistique diminuent avec la complexité de la base. Cette observation nous a permis d'ordonner nos bases suivant le problème qu'elles posent en reconnaissance statistique.

La deuxième partie de ce mémoire décrit notre chaîne de reconnaissance graphique probabiliste avec apprentissage.

Nous commençons, dans le chapitre cinq, par étudier la façon de décrire structurellement une forme. Dans le cadre de la reconnaissance de caractères, la description passe par une étape de squelettisation, une étape d'extraction de points singuliers et de primitives et par un choix de la structure adaptée. Nous avons commencé par revoir différentes méthodes de squelettisation pour valider empiriquement le choix de l'algorithme que nous avons retenu. Ensuite nous avons explicité notre méthode d'extraction de primitives basée sur la détection des points de fin, d'intersection et d'inflexion du squelette. Pour la détection des points d'inflexion nous avons proposé une paramétrisation des primitives à l'aide des courbes de de Casteljau reposant sur des points de contrôle issus d'une étape de polygonalisation afin de lisser la courbe. Enfin nous avons redonné la définition de trois structures classiques et expliqué notre choix pour les graphes d'attributs.

Nous avons ensuite, dans le sixième chapitre, proposé un système de reconnaissance basé sur le formalisme des graphes aléatoires. Le système repose sur une phase d'apprentissage utilisant, au choix, deux algorithmes de recherche d'isomorphismes approximatifs de sous-graphes dont nous rappelons les formalismes ainsi que les fondements scientifiques. Il nous est apparu important d'insister sur deux aspects de la phase d'apprentissage, premièrement le choix des modèles des classes comme les graphes de plus petit ordre et, deuxièmement, le principe de l'appariement comme étant une recherche de clique de poids maximal dans le graphe d'association généralisé. Ensuite nous décrivons la phase de reconnaissance à partir d'une adaptation des graphes aléatoires discrets à notre problématique utilisant des graphes d'attributs continus. Nous proposons de modéliser les éléments aléatoires des graphes par des gaussiennes multivariées et d'utiliser la probabilité de dispersion comme mesure de comparaison entre un graphe d'attributs et les graphes aléatoires ainsi redéfinis. Nous exposons, enfin, notre stratégie de décision en mettant en avant les hypothèses d'indépendances dont elle se sert.

Le septième chapitre est consacré aux résultats de reconnaissance obtenus avec notre système probabiliste de reconnaissance structurelle seul et en coopération avec les approches statistiques de la première partie. Les tests sont réalisés sur les trois bases de caractères à vocabulaire restreint. Nous mettons en avant deux grandes limitations du système structurel proposé : les erreurs d'appariement et la variabilité importante qu'ils impliquent sur les attributs aléatoires. Néanmoins quelques points positifs sont à souligner. Premièrement, l'ordonnancement des bases suivant leur complexité, établi avec les systèmes de reconnaissance statistique, n'est pas retrouvé avec l'approche structurelle ce qui nous permet de conclure que le système apporte une information non redondante et n'ayant pas les mêmes limitations. Cette constation est renforcée par, deuxièmement, les résultats obtenus en coopération. Bien que l'apport de

l'approche structurelle soit encore minime, nous observons qu'il est réel et surtout complémentaire.

Enfin la troisième partie de ce mémoire est consacrée à la présentation d'un système de reconnaissance générique sans apprentissage reposant sur une description des formes par le nouveau concept de graphes d'attributs hiérarchiques flous.

Dans le huitième chapitre nous nous attardons tout d'abord sur le formalisme de la logique floue et, au travers de deux exemples, sur son apport en reconnaissance de caractères. Nous définissons ensuite formellement le concept de graphe d'attributs hiérarchiques flous (GAHF) en le mettant en parallèle avec celui de graphe d'attributs flous. Nous expliquons alors les descriptions utilisées dans notre système ainsi que les mesures de comparaison en redonnant le formalisme des $\mathcal{M}$-mesures de similitude de BOUCHON-MEUNIER ET COLL. ([BMRB96]) qui permet de généraliser nos choix. Enfin nous proposons le schéma de reconnaissance basé sur la description par GAHF qui constitue le cœur de notre système.

Le neuvième et dernier chapitre est consacré aux résultats de reconnaissance obtenus avec notre système flou. Nous mettons en évidence le caractère générique du système basé sur le choix de la mesure de compraison ainsi que l'intérêt de la hiérarchie des attributs. Les limitations de l'approche sont mises en évidence et particulièrement le problème du bruit des structures et du temps de calcul.

10.2 Conclusion générale

Ce travail se veut avant tout prospectif et il est entendu que les systèmes proposés ne sont pas directement utilisables dans un logiciel commercial. Néanmoins nous avons tenté, par la voie de la description graphique, de proposer des solutions aux problèmes actuels de la reconnaissance de caractères manuscrits. Les résultats sont intéressants surtout parce qu'ils valident l'intérêt de ce type d'information sur les bases complexes et pour la réalisation de systèmes génériques. Nous avons proposé des solutions aux différents problèmes techniques rencontrés au cours de la mise au point des deux systèmes. Ces solutions sont bien évidemment imparfaites et nous avons tenté dans la mesure de nos possibilités de proposer des formalismes pouvant servir de base à la recherche d'autres solutions.

10.3 Perspectives

Les perspectives à ces travaux sont nombreuses. Nous proposons de les développer suivant trois axes : la structuration des caractères, l'amélioration des systèmes proposés et les applications possibles.

Concernant la structuration des caractères, il y a à notre avis deux aspects à approfondir. Tout d'abord il faudrait évaluer l'influence de différents prétraitements sur la reconnaissance. Une question ouverte est de savoir s'il ne serait pas préférable de redresser tous les caractères avant le traitement et d'utiliser alors des descriptions plus strictes qui ne soient pas invariantes par rotations. De la même manière ne vaudrait-il pas mieux normaliser la taille des images et s'affranchir de l'invariance par translation ? Le deuxième aspect à approfondir concerne l'extraction des primitives. Notre approche était volontairement focalisée sur l'extraction des primitives réelles. Mais il nous semble après coup qu'il serait intéressant de guider l'extraction des primitives par une modélisation qui permette la suppression ou la fusion des primitives non discriminantes. C'est l'idée qui nous a orienté vers les pyramides combinatoires dans le cadre du stage de Guillaume Lebrun ([Leb05]). Malheureusement ces structures ne semblent pas, pour l'instant, adaptées aux formes squeletisées.

Les deux systèmes de reconnaissance graphique proposés sont encore loin d'être optimisés et il y a particulièrement certains points à reprendre. Concernant le système probabiliste, la limitation principale réside dans le choix de la loi de probabilité pour les éléments aléatoires. Nous avons utilisé le modèle gaussien multivarié mais peut-être serait-ce intéressant d'essayer de paramétriser les distributions et d'utiliser par exemple des distances probabilistes pour apparier les caractères inconnus. Pour le système flou, il faut essayer d'autres mesures de comparaison et trouver un critère simple de sélection de la mesure en fonction de la base. Pour les deux systèmes, enfin, le choix du modèle des classes peut être remis en question en gardant à l'esprit le coup calculatoire et l'imprécision qu'il induit.

La reconnaissance de caractère n'est pas la seule application possible pour ces systèmes graphiques et particulièrement pour le système à base de GAHF. Il est évident que la structure reste valable quelle que soit l'application, ce sont les attributs qu'il convient alors de faire évoluer. De nombreux systèmes à base de graphes sont d'ailleurs déjà utilisés en indexation d'images ou en segmentation multi-échelles. L'intérêt du GAHF est qu'il introduit une hiérarchie dans la description sémantique d'un objet et il pourrait, à ce titre, être intéressant de l'utiliser dans une pyramide de graphes pour la description d'images.

ANNEXE A

CONCEPTION DES BASES DE CARACTÈRES

Sommaire

Les résultats présentés dans ce mémoire ont été obtenus à partir de six bases de caractères et de mots très différentes de par le type, la taille et la nature des informations qu'elles contiennent. L'objectif était de couvrir un large spectre de complexité (figure 1.3 page 4). Une grande partie des applications présentées en introduction de ce mémoire a été adressée. Malheureusement nous n'avons pas réussi à tester nos méthodes sur une base de caractères chinois, à notre grand regret.

A.1 Les bases de caractères

Trois bases de caractères ont été retenues pour les tests. Elles sont relativement différentes, tant par le contenu que par leurs caractéristiques.

A.1.1 Base grecque Coroc : GrCor

Cette base de caractères a été réalisée au sein de RC-Soft pour les tests internes.

A.1.1.1 Acquisition et prétraitements

Les caractères ont été écrits sur des pages blanches avec un feutre noir par quatre scripteurs différents. L'acquisition a été effectuée avec un scanner grand public à 300 dpi. Les images étant de bonne qualité, les prétraitements se sont résumés à une binarisation manuelle. La segmentation a été réalisée par une analyse en composantes connexes avec calcul de boîtes englobantes. Enfin, deux petits postraitements ont été effectués. Les boîtes englobantes ayant une intersection non nulle ont été fusionnées et un seuillage sur la taille des boîtes a été opéré. Les boîtes restantes composent les caractères de la base d'apprentissage. L'indexation est automatique puisque chaque page d'écriture correspond à un caractère donné.

A.1.1.2 Caractéristiques

La base possède 47 classes pour un total de 18 236 caractères. La table A.1 donne leur répartition. Lorsque le nom d'une classe se termine par « -M » c'est qu'il s'agit d'un caractère majuscule. Les majuscules n'ont été écrites que par un seul scripteur et sont donc en nombre inférieur. Nous les avons intégrées à la base car elles posent des problèmes de différenciation avec les minuscules.

Classe	alp	alp-M	bet	bet-M	chi	chi-M	del	del-M	dze	eps
Nombre	822	65	810	65	615	68	845	60	680	774
Classe	eps-M	eta	eta-M	gam	gam-M	iot	iot-M	kap	kap-M	lam
Nombre	54	812	46	846	58	1010	65	580	79	592
Classe	lam-M	mu	mu-M	nu	nu-M	ome	ome-M	omi	omi-M	phi
Nombre	52	583	45	609	84	572	74	671	56	547
Classe	phi-M	pi	pi-M	psi	psi-M	rho	rho-M	sig	sig-M	sig2
Nombre	51	563	53	549	58	621	74	666	45	590
Classe	the	to	to-M	ups	ups-M	xi	zet-M			
Nombre	902	572	56	590	44	518	45			

TABLE A.1 – Contenu de la base GrCor

Les modèles, sélectionnés automatiquement comme étant les caractères dont le squelette possède le moins de points singuliers, sont présentés dans la table A.2 page ci-contre.

A.1.2 Base de caractères grecs anciens : GrAnc

C'est une base restreinte de caractères grecs issus de documents manuscrits anciens du Moyen-Age.

A.1.2.1 Acquisition et prétraitements

Les caractères sont issus de quatre pages de manuscrits grecs anciens différents et d'époques différentes (présentées dans la table A.1 page 158). L'acquisition a été effectuée avec un scanner grand public à 300 dpi. Les images ont été binarisées puis ont subi une fermeture morphologique pour supprimer les pixels isolés. La segmentation a été réalisée par une analyse en composantes connexes avec calcul de boîtes englobantes. Comme pour la base précédente, les boîtes englobantes ayant une intersection non nulle ont été fusionnées et un seuillage sur la taille des boîtes a été opéré. Les boîtes restantes ont été

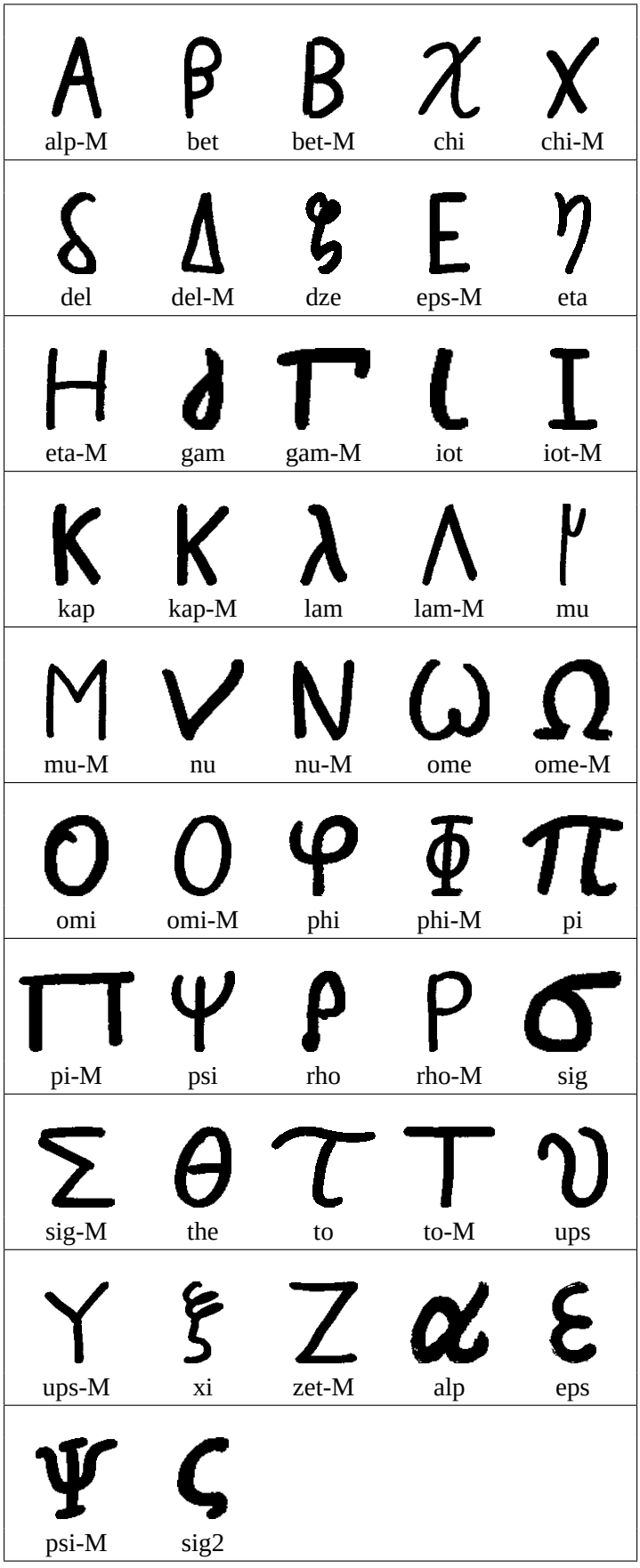


TABLE A.2 – Modèles de la base GrCor.

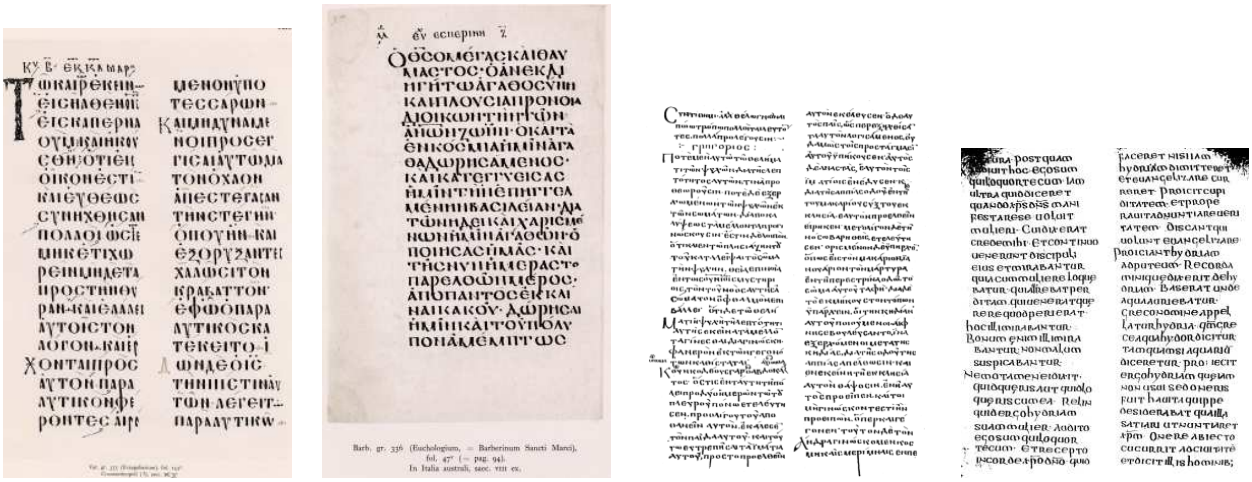


FIGURE A.1 – Pages de grec ayant servi à la constitution de la base GrAnc.

triées et indexées à la main pour réaliser la base d'apprentissage. Avec ce procédé les caractères retenus ne sont pas accentués.

A.1.2.2 Caractéristiques

La base possède 38 classes pour un total de 1 445 caractères. La table A.3 donne leur répartition. Elle contient relativement peu d'éléments et la répartition suivant les classes est très hétérogène.

Classe	am	b	c	d	epb	eps	f	g	h	hm
Nombre	29	12	56	23	4	152	1	5	62	5
Classe	i	k	kap	l	lam	ldb	m	mm	n	nb
Nombre	76	3	84	12	138	2	21	29	127	27
Classe	nu	o	ome	p	phi	phm	pi	q	r	s
Nombre	54	71	22	36	6	1	49	14	41	13
Classe	t	tet	to	u	x	y	z	zet		
Nombre	164	19	20	53	6	3	1	4		

TABLE A.3 – Contenu de la base GrAnc

Les modèles, sélectionnés automatiquement comme étant les caractères dont le squelette possède le moins de points singuliers, sont présentés dans la table A.4 page suivante.

A.1.3 Base de chiffres manuscrits : MNIST

Nous avons également utilisé la base de chiffres manuscrits MNIST[1], qui a servi à valider plusieurs travaux de reconnaissance [LBBH98], afin de comparer nos méthodes à l'existant.

A.1.3.1 Acquisition et prétraitements

Les caractères se présentent sous forme d'images en niveaux de gris de 28×28 pixels. Nous n'avons utilisé que la base de test composée de 10 000 images provenant pour moitié d'employés de l'US Census

[1] http ://yann.lecun.com/exdb/mnist/

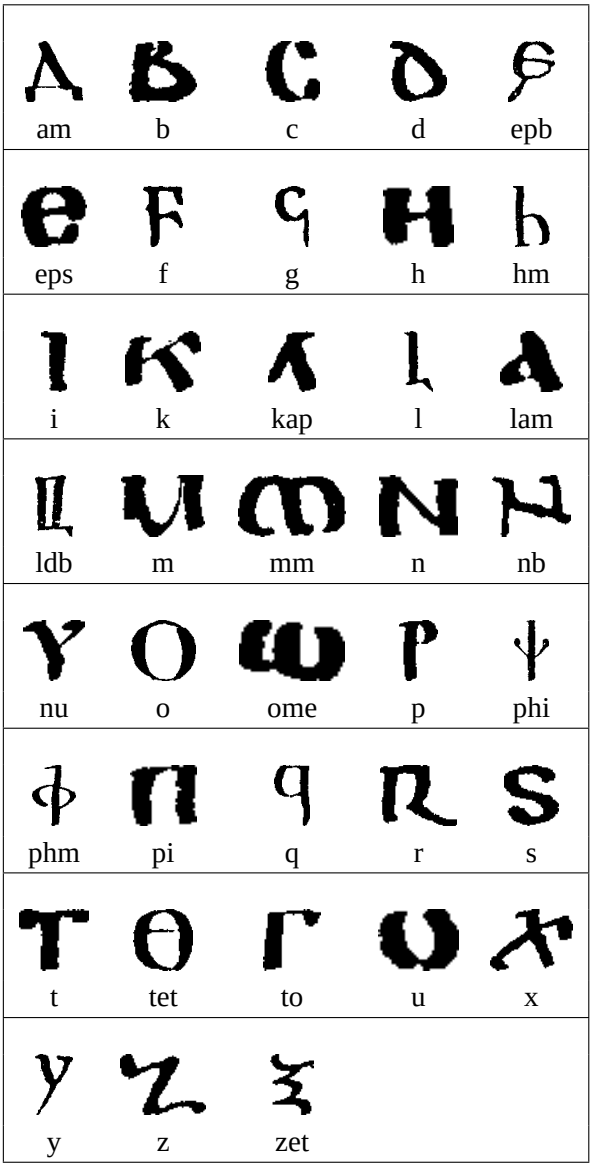


TABLE A.4 – Modèles de la base GrAnc.

Classe	0	1	2	3	4
Nombre	980	1135	1032	1010	982
Classe	5	6	7	8	9
Nombre	892	958	1028	974	1009

TABLE A.5 – Contenu de la base MNIST

Bureau (Office américain chargé du recensement de la population) et pour moitié d'étudiants du supérieur. Ils ont été réalisés par environ 250 scripteurs différents. Aucun traitement supplémentaire n'a été réalisé sur ces images excepté une binarisation.

A.1.3.2 Caractéristiques

La base comporte 10 classes de chiffres (de 0 à 9) dénombrés dans la table A.5. Chaque classe est bien fournie et toutes les classes sont homogènes.

 980	 1009	 1135	 958	 982
 1032	 892	 1010	 1028	 974

TABLE A.6 – Modèles de la base MNIST avec le nombre d'éléments de la classe associée.

Les modèles sélectionnés sont présentés dans la table A.6 ainsi que le nombre d'éléments de la classe qu'ils représentent.

A.2 La base de mots arabes

Nous avons utilisé la base de noms de villes tunisiennes arabes de l'IFN/ENIT[2] pour les tests structurels à base de GAHF essentiellement. La version finale de la base retenue compte 9 534 images classées par le code postal de la ville.

A.2.1 Acquisition et prétraitements

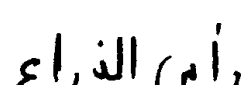
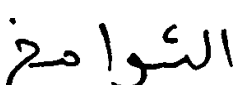
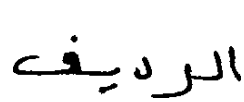
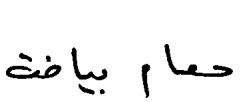
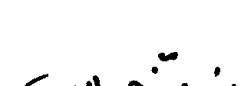
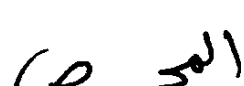
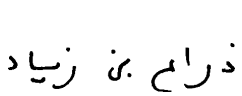
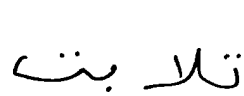
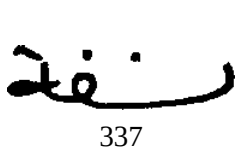
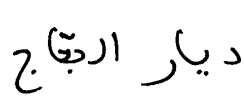
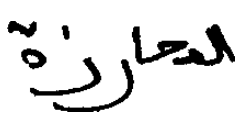
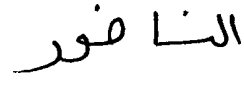
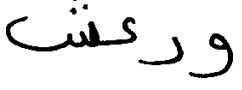
Nous avons utilisé les images telles qu'elles sont fournies avec la base acquise à 300dpi. Un tri a été effectué sur la taille de l'image afin d'enlever un maximum de noms de ville mal segmentés. Ensuite pour pouvoir tester les méthodes par GAHF dans un temps raisonnable, nous n'avons gardé que les classes dont au moins un caractère, une fois squelettisé (voir figure 5.25 page 85), possédait moins de 15 points singuliers.

A.2.2 Caractéristiques

Au final, notre base est composée de 78 classes provenant de 411 scripteurs différents. La table A.7 page suivante présente tous les modèles des classes avec le nombre d'individus de même étiquette dans la base.

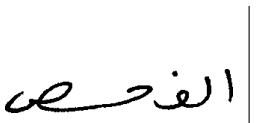
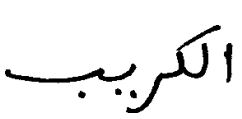
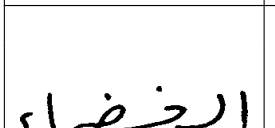
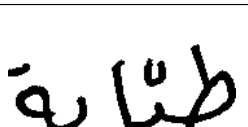
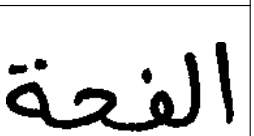
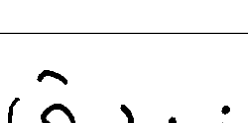
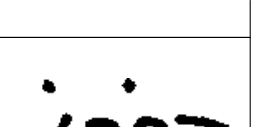
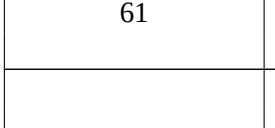
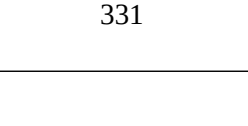
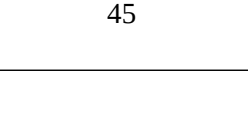
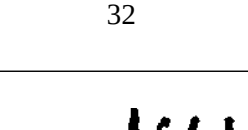
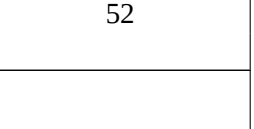
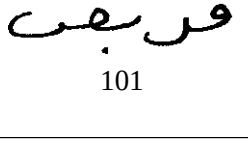
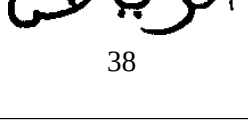
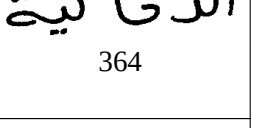
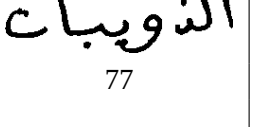
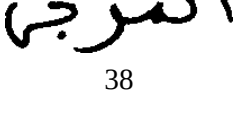
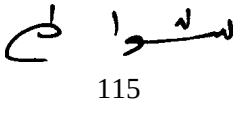
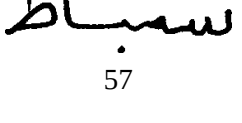
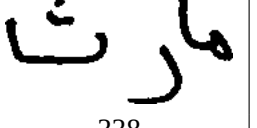
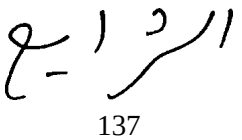
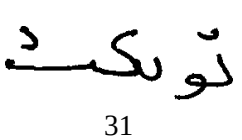
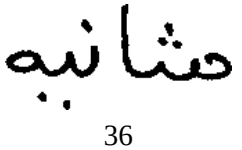
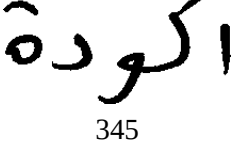
[2]http ://www.ifnenit.com/

TABLE A.7: Modèles de la base IFN/ENIT.

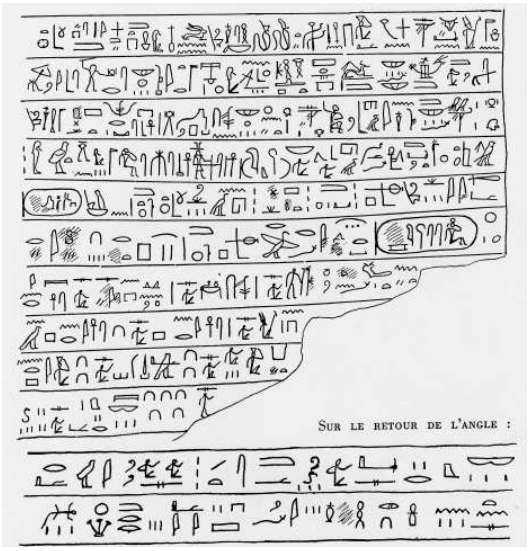
35	 309	 74	 368	 132
 350	 52	 94	 31	 99
 62	 348	 37	 83	 82
 38	 84	 381	 37	 44
 85	 38	 337	 340	 173
 59	 43	 311	 138	 77
 42	 66	 47	 67	 35

Suite à la page suivante

TABLE A.7: suite

 160	 38	 29	 368	 78
 73	 33	 38	 33	 52
 61	331	 45	32	 52
 50	 101	 38	 346	 364
 77	 38	 115	 57	 344
 338	 137	 31	 36	 345
 69	 36	 342	 96	 32

Suite à la page suivante

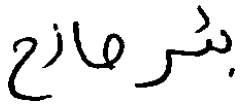
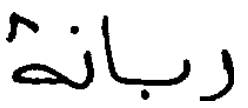
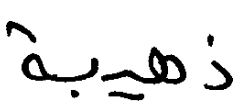
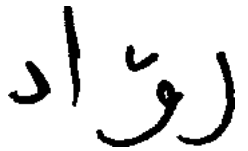
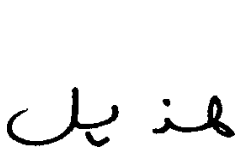
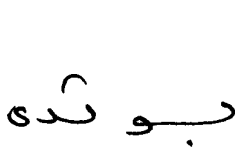


Page de modèles

Page de tests

FIGURE A.2 – Pages de hiéroglyphes ayant servi aux tests. La première page a été utilisée pour construire la base et la deuxième pour extraire des caractères a indexer.

TABLE A.7: suite

 37	 78	 41	 68	 68
 72	 90	 67		

A.3 La base de hiéroglyphes manuscrits : HrMan

Cette base a été réalisée à partir d'une page de manuscrit hiéroglyphique prêtée par un égyptologue. L'indexation manuelle des hiéroglyphes n'étant réalisable que par un expert, elle n'est malheureusement pas indexée et n'a donc pas pu servir pour tester des systèmes de reconnaissance. Chaque caractère est un modèle. Une deuxième page, d'un autre égyptologue (scripteur différent donc) a servi à extraire les caractères à indexer. La figure A.2 présente ces deux pages manuscrites.

A.3.1 Acquisition et prétraitements

Les caractères modèles sont présentés dans la table. L'acquisition a été effectuée avec un scanner grand public à 300 dpi. Les images ont été binarisées puis ont subi une fermeture morphologique pour supprimer les pixels isolés. La segmentation a été réalisée par une analyse en composantes connexes avec calcul de boîtes englobantes. Les boîtes ont été triées à la main pour réaliser la base d'apprentissage.

A.3.1.1 Caractéristiques

La base possède 94 caractères. Ils ont été sélectionnés pour leur bonne segmentation et non pour leur contenu et plusieurs caractères peuvent posséder le même contenu sémantique.

TABLE A.8: Caractères hiéroglyphiques.

ANNEXE B

ERREUR BAYÉSIENNE

La probabilité d'erreur bayésienne moyenne (appelée couramment *erreur bayésienne*) est la probabilité globale qu'un estimateur bayésien donne une estimation fausse. Elle est obligatoirement associée à une fonction de coût. La fonction de coût donnant la probabilité d'erreur la plus faible sera, sans connaissance *a priori*, la fonction $0-1$.

En classification, la probabilité d'erreur d'associer une observation $\mathbf{y}$ à une classe par ϵ s'exprime de la façon suivante :

$$Pr(\epsilon(\mathbf{y})) = E(\mathcal{R}(\epsilon(\mathbf{y}), \mathbf{y})) = \int_{C_\epsilon} \mathcal{R}(\epsilon(\mathbf{y}), \mathbf{y}) f(\mathbf{y}) d\mathbf{y} \tag{B.1}$$

où C_ϵ est la région de $\mathbb{R}^n$ où $\epsilon(\mathbf{y})$ maximise la probabilité *a posteriori*, c'est-à-dire :

$$\forall \mathbf{y} \in C_\epsilon,\ \epsilon(\mathbf{y}) = \arg\max_{\omega_i} p(\omega_i|\mathbf{y}) \tag{B.2}$$

où $\mathcal{R}(\epsilon(\mathbf{y}), \mathbf{y})$ est le risque bayésien[1] associé à l'estimation $\epsilon(\mathbf{y})$, il s'écrit :

$$\mathcal{R}(\epsilon(\mathbf{y}), \mathbf{y}) = \sum_{j=1}^{c} l(\epsilon(\mathbf{y}), \omega_j) p(\omega_j|\mathbf{y}) = 1 - p(\epsilon(\mathbf{y})|\mathbf{y}) \text{ avec la fonction de coût } 0-1 \tag{B.3}$$

La probabilité d'erreur de l'estimation $\epsilon(\mathbf{y})$ s'écrit donc :

$$Pr(\epsilon(\mathbf{y})) = \int_{C_\epsilon} f(\mathbf{y}) d\mathbf{y} - \int_{C_\epsilon} p(\epsilon(\mathbf{y})|\mathbf{y}) f(\mathbf{y}) d\mathbf{y} \tag{B.4}$$

L'erreur bayésienne est la somme des erreurs pour chaque estimation :

$$\begin{aligned} Pr_{bayes} = \sum_{i=1}^{c} Pr(\epsilon(\mathbf{y}) = \omega_i) \quad &= \sum_{i=1}^{c} \left(\int_{C_\epsilon} f(\mathbf{y}) d\mathbf{y} - \int_{C_\epsilon} p(\epsilon(\mathbf{y}) = \omega_i|\mathbf{y}) f(\mathbf{y}) d\mathbf{y} \right) \\ &= \underbrace{\sum_{i=1}^{c} \left(\int_{C_i} f(\mathbf{y}) d\mathbf{y} \right)}_{1} - \sum_{i=1}^{c} \left(\int_{C_i} p(\omega_i|\mathbf{y}) f(\mathbf{y}) d\mathbf{y} \right) \\ &= 1 - \sum_{i=1}^{c} \left(\int_{C_i} p(\omega_i|\mathbf{y}) f(\mathbf{y}) d\mathbf{y} \right) \\ &= 1 - \sum_{i=1}^{c} \left(\int_{C_i} f(\mathbf{y}|\omega_i) p(\omega_i) d\mathbf{y} \right) \end{aligned} \tag{B.5}$$

où les C_i sont les régions de $\mathbb{R}^n$ où $\epsilon(\mathbf{y}) = \omega_i$ maximise la probabilité a postériori.

[1] nom justifié par le fait que $\epsilon(\mathbf{y}) = \arg\min_{\omega_i}(\mathcal{R}(\omega_i, \mathbf{y}))$

ANNEXE C

MORPHOLOGIE MATHÉMATIQUE POUR DES IMAGES BINAIRES

Sommaire

L'approche morphologique consiste à ne plus considérer l'image comme un ensemble de points d'un espace vectoriel mais comme un ensemble dont la structure fondamentale est le treillis formé par les pixels. C'est pourquoi les algorithmes de morphologie s'appliquent à l'origine à des images binaires (même s'il existe maintenant des extensions pour les images en niveaux de gris et même en couleur) où l'on ne considère que l'appartenance des points du treillis à un ensemble donné (1 pour l'appartenance et 0 pour la non appartenance).

Voici un rappel des opérations classiques de la morphologie mathématique sur des images binaires. Pour un aperçu plus approfondi, nous recommandons la lecture du livre de SCHMITT ET MATTIOLI ([SM93]).

C.1 Les opérations de base

C.1.1 L'érosion

Soit B un élément structurant de dimension impaire et B_x cet élément centré en un pixel x. L'érosion de X par B consiste à poser en chaque pixel x de l'image, la question : « B_x est-il contenu entièrement dans X ? ». L'ensemble des positions x correspondant à une réponse positive forme le nouvel ensemble Y, appelé *érodé* de X par B. Cet ensemble satisfait l'équation :

$$Y = X \ominus B = \{x | B_x \subseteq X\} \tag{C.1}$$

C.1.2 La dilatation

L'opération de dilatation se définit de manière analogue à l'érosion. En prenant le même élément structurant centré en x, B_x, la question posée pour chaque point x de l'image est : « B_x touche-t-il l'ensemble X ? ». C'est à dire, existe-t'il une intersection non vide entre B_x et X ?

L'ensemble des points x de l'image correspondant aux réponses positives forme le nouvel ensemble X des *dilatés* de X.

$$Y = X \oplus B = \{x | B_x \cap X \neq \oslash\} \tag{C.2}$$

C.1.3 L'ouverture

L'ouverture est l'application de l'opérateur érosion puis de l'opérateur dilatation avec le même élément structurant. Y, l'ouverture de X par B, est notée :

$$Y = X \circ B = (X \ominus B) \oplus B \tag{C.3}$$

En général, l'ensemble de départ n'est pas conservé car une partie de la forme éliminée par l'érosion ne peut être recréée par une dilatation. L'ensemble Y est plus régulier (moins de détails au niveau du contour) que l'ensemble initial X. En termes « géographiques » ou morphologiques, l'ouverture adoucit les contours, coupe les isthmes étroits, supprime les petites îles et les caps étroits.

C.1.4 La fermeture

La fermeture est l'opération « inverse » de l'ouverture. Elle consiste en une dilatation suivie d'une érosion (toujours en gardant le même élément structurant) :

$$Y = X \bullet B = (X \oplus B) \ominus B \tag{C.4}$$

Un ensemble fermé est également moins riche en détails que l'ensemble initial. La transformation par fermeture bouche les canaux étroits, supprime les petits lacs et les golfes étroits.

Voici quelques propriétés :
- Les deux opérations d'ouverture et de fermeture sont *idempotentes*, c'est à dire que le résultat est invariant après itérations.
- Ces transformations ne sont pas *homotopiques* (c'est à dire qu'elles ne préservent pas la connexité) car elles peuvent scinder une forme en deux (érosion) ou fusionner deux formes (dilatation). Elles ne présentent donc pas de propriétés topologiques intéressantes. Ce ne sont que des transformations d'aspect, de simplification ou de filtrage.

C.2 Quelques algorithmes élémentaires

C.2.1 Extraction du contour

Le contour interne d'un objet s'obtient par une érosion d'un objet A (suivant l'élément B) suivie d'une différence :

$$A - (A \ominus B) \tag{C.5}$$

Le contour externe s'obtient par une dilatation suivie d'une différence :

$$A - (A \oplus B) \tag{C.6}$$

C.2.2 Transformation tout ou rien

Cette opération permet de trouver des configurations dans une image (un angle par exemple). La présence de B dans A sera attestée par un point suivant la transformation suivante :

$$A \circledast B = (A \ominus B) \cap (A^C \ominus B^C) \tag{C.7}$$

où A^C désigne le complémentaire de A. Typiquement, pour la recherche d'un angle, l'élément structurant du type :

$$B = \begin{bmatrix} \times & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{C.8}$$

où $\times$ signifie que la valeur de cet élément est indifférente. Prenons l'exemple où il est remplacé par 0 alors :

$$A \circledast B = (A \ominus \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}) \cap (A^C \ominus \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}) \tag{C.9}$$

Un autre exemple est illustré à la figure C.1.

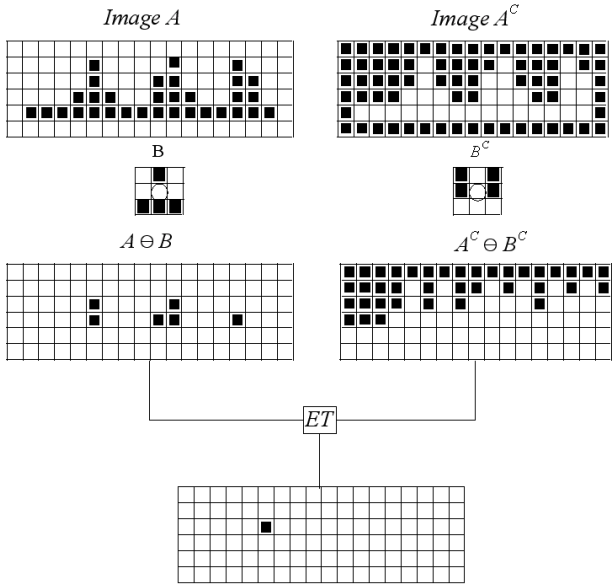


FIGURE C.1 – Exemple de transformation tout ou rien (les problèmes de bords sont gérés en ajoutant une ligne blanche en haut et en bas ainsi qu'une colonne blanche à gauche et à droite pendant les calculs).

C.2.3 Ensemble convexe circonscrit - approche région

Par cet algorithme, le polygone convexe le plus petit contenant l'objet A est recherché. Pour ce faire les opérations suivantes sont utilisées en notant C(A) le polygone final :

$$C(A) = \cup_{i=1}^{n} C_i(A) \cup A \tag{C.10}$$

où :

$$C_i(A) = \cup_{k=1}^{K} C_i^{(k)}(A) \tag{C.11}$$

avec :

$$C_i^{(k)}(A) = C_i^{(k-1)}(A) \circledast B_i \quad et \quad C_i^{(0)} = A \tag{C.12}$$

K est le nombre maximum d'opérations tout ou rien permettant d'obtenir une image non vide, les B_i sont les éléments structurants qui vont définir le polygone et $\cup_{k=1}^{K}$ signifie l'union de toutes les images calculées successivement. Pour les B_i, dans le cas d'un parallélogramme avec les côtés inclinés à 45° par rapport au cadre de l'image, les masques suivants sont utilisés :

$$B_1 = \begin{bmatrix} 1 & \times & \times \\ 1 & 0 & \times \\ 1 & \times & \times \end{bmatrix} \tag{C.13}$$

$$B_2 = \begin{bmatrix} 1 & 1 & 1 \\ \times & 0 & \times \\ \times & \times & \times \end{bmatrix} \tag{C.14}$$

$$B_3 = \begin{bmatrix} \times & \times & 1 \\ \times & 0 & 1 \\ \times & \times & 1 \end{bmatrix} \tag{C.15}$$

$$B_4 = \begin{bmatrix} \times & \times & \times \\ \times & 0 & \times \\ 1 & 1 & 1 \end{bmatrix} \tag{C.16}$$

Le résultat est présenté à la figure C.2.

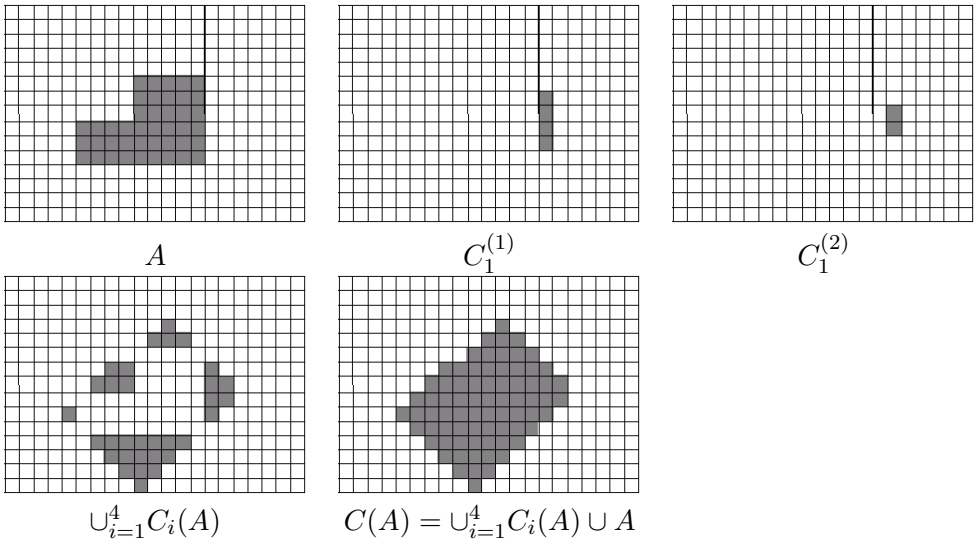


FIGURE C.2 – Construction du paraléllogramme à 45° englobant

Il y a d'autres méthodes pour obtenir d'autres boîtes englobantes, soit avec d'autres éléments B_i et d'autres opérations morphologiques, soit avec des approches différentes. Les éléments de la figure C.3 page suivante par exemple, donnent des polygones complexes très intéressants.

C.2.4 Amincissement de GONZALES ET WOOD ([GW92])

GONZALES ET WOOD ([GW92]) propose une implémentation morphologique de leur algorithme d'amincissement reposant sur l'opération suivante :

$$A \otimes B = A - (A \circledast B) \tag{C.17}$$

$$B_1 = \begin{bmatrix} 1 & 1 & \times \\ 1 & 0 & \times \\ 1 & \times & \times \end{bmatrix}, B_2 = \begin{bmatrix} 1 & 1 & 1 \\ \times & 0 & 1 \\ \times & \times & \times \end{bmatrix}, B_3 = \begin{bmatrix} \times & \times & 1 \\ \times & 0 & 1 \\ \times & 1 & 1 \end{bmatrix}, B_4 = \begin{bmatrix} \times & \times & \times \\ 1 & 0 & \times \\ 1 & 1 & 1 \end{bmatrix}$$
$$B_5 = \begin{bmatrix} \times & 1 & 1 \\ \times & 0 & 1 \\ \times & \times & 1 \end{bmatrix}, B_6 = \begin{bmatrix} \times & \times & \times \\ \times & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}, B_7 = \begin{bmatrix} 1 & \times & \times \\ 1 & 0 & \times \\ 1 & 1 & \times \end{bmatrix}, B_8 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & \times \\ \times & \times & \times \end{bmatrix}$$

FIGURE C.3 – exemple d'éléments structurants pour définir une boîte englobante

B n'est pas un masque unique mais la séquence d'éléments structurants suivante :

$$\{B\} = \{B^1, B^2, B^3, ..., B^n\} \tag{C.18}$$

où B^i est la rotation d'un pixel (vers la droite ou la gauche suivant la convention choisie) de B^{i-1}. Alors l'opération d'amincissement devient :

$$A \otimes \{B\} = ((...((A \circledast B^1) \circledast B^2)...) \circledast B^n) \tag{C.19}$$

Et ce processus est répété jusqu'à ce que plus aucun changement n'affecte la forme. Les éléments structurants proposés par GONZALES ET WOOD sont les suivants :

$$B^1 = \begin{bmatrix} 0 & 0 & 0 \\ \times & 1 & \times \\ 1 & 1 & 1 \end{bmatrix}, \quad B^2 = \begin{bmatrix} \times & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & \times \end{bmatrix}, \quad B^3 = \begin{bmatrix} 1 & \times & 0 \\ 1 & 1 & 0 \\ 1 & \times & 0 \end{bmatrix}, ..., \quad B^8 = \begin{bmatrix} 0 & 0 & \times \\ 0 & 1 & 1 \\ \times & 1 & 1 \end{bmatrix} \tag{C.20}$$

C.2.5 Ebarbulage

L'ébarbulage consiste à enlever les segments parasites qui peuvent apparaître à la fin d'un amincissement ou d'une squelettisation. Le principe repose sur 4 opérations. La première consiste à amincir la forme avec des masques d'ébarbulage :

$$X_1 = A \otimes \{B\} \tag{C.21}$$

où $\{B\}$ est la séquence des éléments structurants suivants (représentant des points finaux) :

$$B^1 = \begin{bmatrix} \times & 0 & 0 \\ 1 & 1 & 0 \\ \times & 0 & 0 \end{bmatrix}, B^2, \quad B^3, \quad B^4 \quad \text{par rotation de 90°.}$$
$$et \tag{C.22}$$
$$B^5 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, B^6, \quad B^7, \quad B^8 \quad \text{par rotation de 90°.}$$

Cette opération est réalisée un nombre de fois prédéterminé en fonction de la taille des segments à ébarbuler. Les trois étapes suivantes vont alors consister à « restaurer » la forme originale mais nettoyée des éléments parasites. La première opération va marquer tous les points finaux de l'image résultante X_1 :

$$X_2 = \bigcup_{k=1}^{8} (X_1 \circledast B^k) \tag{C.23}$$

La deuxième opération va dilater ces points autant de fois que la forme a été amincie par l'équation C.21) :

$$X_3 = (X_2 \oplus H) \cap A \tag{C.24}$$

où H est un masque structurant 3×3 plein. Et enfin la troisième opération réalise l'union de X_1 et X_3 pour donner le résultat final :

$$X_4 = X_1 \cup X_3 \tag{C.25}$$

ANNEXE D

LA LOGIQUE FLOUE

Sommaire

La logique floue est née suite à la constatation que la logique classique, considérant les évènements uniquements comme étant vrais ou faux, était trop limitée pour modéliser des raisonnements humains plus vagues. Un exemple couramment utilisé est celui de la taille d'un individu. A partir de quand est-il possible de qualifier quelqu'un de grand ou petit ? La logique classique définira des seuils fixes et posera arbitrairement qu'au dessus de 1m80 un homme est considéré comme grand et qu'en-dessous de 1m50 il est considéré comme petit. Ce raisonnement est efficace mais limité (si je fais 1m80, suis-je grand ?) alors que le raisonnement humain est bien plus nuancé que cela. La logique classique peut même poser des problèmes d'incompréhension majeurs entre homme et machine par exemple dans l'interrogation d'une base de donnée avec des termes flous comme « un homme grand ». L'ordinateur doit-il rejeter les personnes de 1m79 pour cette requête ? Pour répondre à ce problème, ZADEH ([Zad65]) introduit en 1965 les concepts fondamentaux de la logique floue. Il considère la notion d'appartenance d'un objet à un ensemble non plus comme une fonction booléenne mais comme une fonction qui peut prendre toutes les valeurs entre 0 et 1.

Nous redonnons ici quelques notations et définitions utiles en logique floue. Pour plus de détail, le lecteur pourra se reporter à l'ouvrage de GACÔGNE ([Gac97]).

D.1 Théorie des sous-ensembles flous

D.1.1 Définitions

Un sous-ensemble flou A d'un ensemble de référence E est caractérisé par une fonction d'appartenance $f_A : E \rightarrow [0,1]$. Pour un élément $x \in E$, $f_A(x)$ est notée $\mu_A(x)$ et représente le degré d'appartenance de x à A. Nous confondrons par commodité ces deux écritures. Voici un rappel de quelques définitions fondamentales en logique floue.

Définition 43 (Noyau) *Le noyau de A est défini par $N(A) = \{x|\mu_A(x) = 1\}$, il constitue l'ensemble des éléments « vraiment » dans A.*

Définition 44 (Support) *Le support de A est définit par $S(A) = \{x|\mu_A(x) \neq 0\}$, il constitue l'ensemble des élément « plus ou moins » dans A.*

Dans $\mathbb{R}$ les fonctions d'appartenance peuvent avoir plusieurs formes dont la plus courante est le trapèze (figure D.1).

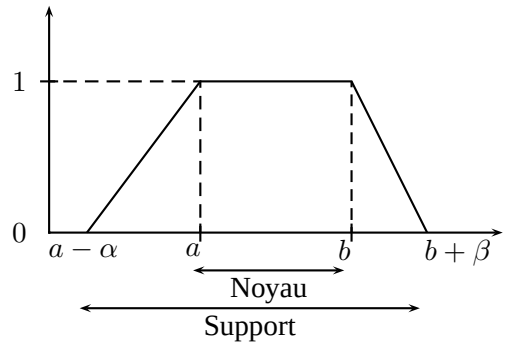


FIGURE D.1 – Fonction d'appartenance trapézoïdale

Définition 45 (Hauteur) $ht(A) = sup_{x \in A}\mu_A(x)$ *est la hauteur de A, il s'agit de la borne supérieure de la fonction d'appartenance.*

Définition 46 (Cardinalité) $|A| = \sum_{x \in E} \mu_A(x)$, *la cardinalité de A est la quantité floue d'éléments de E qui appartiennent à A.*

Définition 47 (α-coupe) *Une α-coupe de A est un sous-ensemble de E tel que $A_\alpha = \{x \in E|\mu_A(x) \geq \alpha\}$. Il représente l'ensemble des éléments de E qui appartiennent à A avec un dégré supérieur ou égal à α.*

D.1.2 Opérations ensemblistes, définitions originales de ZADEH

La théorie des sous-ensembles flous est une extension de la théorie ensembliste classique. ZADEH ([Zad65]) montre que les opérateurs flous se déduisent des opérateurs classiques par généralisation. Il s'agit des opérateurs ensemblistes flous dits classiques ou au sens de ZADEH. Soient A et B deux sous-ensembles flous de E :

Définition 48 (Egalité) $A = B \iff \forall x \in E,\ \mu_A(x) = \mu_B(x)$

Définition 49 (Inclusion) $A \subseteq B \iff \forall x \in E,\ \mu_A(x) \leq \mu_B(x)$

Définition 50 (Union) $A \cup B = max(\mu_A(x), \mu_B(x)),\ \forall x \in E$

Définition 51 (Intersection) $A \cap B = min(\mu_A(x), \mu_B(x)),\ \forall x \in E$

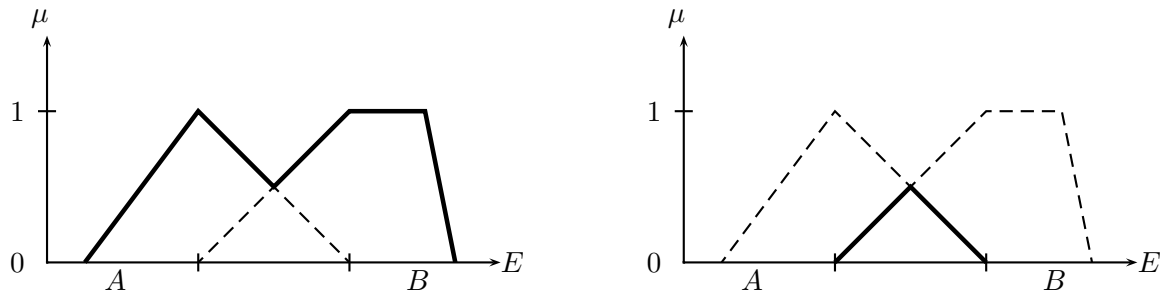


FIGURE D.2 – Fonctions d'appartenances $A \cup B$ et $A \cap B$

Définition 52 (Complémentation) *$\overline{A}$ est dit complémentaire de A si sa fonction d'appartenance vérifie :* $\mu_{\overline{A}}(x) = 1 - \mu_A(x),\ \forall x \in E$

Définition 53 (Produit cartésien) *Si A et B sont deux sous-ensembles flous de E et F respectivement, le produit cartésien $A \times B$ est défini comme un sous-ensemble flou de $E \times F$ dont la fonction d'appartenance est telle que :* $\mu_{A\times B}(x, y) = min\{\mu_A(x), \mu_B(y)\},\ \forall (x, y) \in (E \times F)$

Définition 54 (La projection) *Si C est un sous-ensemble flou de $E \times F$, la projection de C sur E est un sous-ensemble flou A de E dont la fonction d'appartenance se définit par* $\mu_A(x) = \sup_{y\in F}\{\mu_C(x, y)\},\ \forall x \in E$.

D.1.3 Relations floues

ZADEH étend le concept de relations aux sous-ensembles flous ([Zad71]).Une relation floue se définit de manière générale sur deux univers quelconques par :

Définition 55 (Relation floue) *Soit deux univers E_1 et E_2, une relation floue R est un ensemble flou sur $E_1 \times E_2$ défini par sa fonction d'appartenance $\mu_R : E_1 \times E_2 \to [0, 1]$:*

$$R = \{((x, y), \mu_R(x, y))|(x, y) \in E_1 \times E_2\} \tag{D.1}$$

Cette définition s'étend aux sous-ensembles flous. Si A et B sont deux sous-ensembles flous de E_1 et E_2 respectivement, alors une relation floue sur $A \times B$ est définie comme un ensemble flou de $E_1 \times E_2$ tel que :

$$\forall (x, y) \in E_1 \times E_2,\ \mu_R(x, y) \leq \min(\mu_A(x), \mu_B(y)) \tag{D.2}$$

D.1.3.1 Propriétés générales ([BM93])

Définition 56 (Inverse) *L'inverse d'une relation floue R entre E_1 et E_2 est une relation floue entre E_2 et E_1 notée R^{-1} et définie par :*

$$\forall x \in E_1,\ \forall y \in E_2,\ \mu_R^{-1}(y, x) = \mu_R(x, y) \tag{D.3}$$

Définition 57 (Composition max-min) *Une composition max-min de deux relations floues R_1 entre E_1 et E_2 et R_2 entre E_2 et E_3 est une relation floue entre E_1 et E_3 notée $R = R_1 \circ R_2$ et dont la fonction d'appartenance est telle que :*

$$\forall x \in E_1,\ \forall z \in E_3,\ \mu_R(x,z) = \sup_{y \in E_2} \min\left(\mu_{R_1}(x,y), \mu_{R_2}(y,z)\right) \tag{D.4}$$

La composition max-min est la plus classiquement utilisée, mais il est possible de remplacer le min par un autre opérateur *. Il s'agit alors d'une composition max-*.

D.1.3.2 Propriétés particulières ([BM93])

Les propriétés des relations binaires sur $E \times E$ s'étendent aux relations floues.

Définition 58 Réflexivité *Une relation floue sur $E \times E$ est reflexive si :*

$$\forall x \in E,\ \mu_R(x,x) = 1 \tag{D.5}$$

Définition 59 Symétrie *Une relation floue sur $E \times E$ est symétrique si :*

$$\forall (x,y) \in E \times E,\ \mu_R(x,y) = \mu_R(y,x) \tag{D.6}$$

Définition 60 Antisymétrie *Une relation floue sur $E \times E$ est antisymétrique si :*

$$\forall (x,y) \in E \times E,\ (\mu_R(x,y) > 0 \text{ et } \mu_R(y,x) < 0) \Rightarrow x = y \tag{D.7}$$

Définition 61 Transitivité max-min *Une relation floue sur $E \times E$ est transitive max-min si :*

$$\forall (x,y,z) \in E \times E \times E,\ \mu_R(x,z) \geq \sup_{y \in E} \min\left(\mu_R(x,y), \mu_R(y,z)\right) \tag{D.8}$$

Comme pour la composition, l'opérateur min peut être remplacer par un autre opérateur *, il s'agit alors d'une propriété de transitivité max-*.

D.1.3.3 Relations de similarité ([BM93])

Définition 62 (Relation de similarité) *Une relation de similarité est une relation floue R entre E et E qui vérifie les trois propriétés particulières suivantes :*

- *R est réflexive,*
- *R est symétrique,*
- *R est max-min transitive.*

Certaines variantes de relations de similarité utilise une transitivité max-*. La réflexivité peut parfois être difficile à obtenir et il est courant d'utiliser des pseudo-relations de similarité qui ne remplissent pas ou que partiellement (par exemple $\mu_R(x,x) \geq \alpha$) cette condition.

Si une relation de similarité R est telle que $\mu_R(x,y) = 1$ si et seulement si $x = y$ alors il est possible de lui associer une ultramétrique d définie sur E et à valeurs dans $[0,1]$ définie par :

$$\forall (x,y) \in E \times E,\ d(x,y) = 1 - \mu_R(x,y) \tag{D.9}$$

et d vérifie :

$$\forall (x,y,z) \in E \times E \times E,\ d(x,y) \leq \max(d(x,z), d(z,y)) \tag{D.10}$$

Nom	Fonction
Zadeh	$min(x, y)$
Probabiliste	$x.y$
Lukasiewicz	$max(x + y - 1, 0)$
Yager	$1 - min\{1, (1 - (x + y))^{1/\beta}\}$

TABLE D.1 – Exemples de t-normes

Nom	Fonction
Zadeh	$max(x, y)$
Probabiliste	$x + y - x.y$
Lukasiewicz	$min(x + y, 1)$
Yager	$min\{1, (x + y)^{1/\beta}\}$

TABLE D.2 – Exemples de t-conormes

D.1.3.4 Relations d'ordre floues ([BM93])

Une relation floue réflexive et max-min transitive est une *relation de pré-ordre flou*. Si elle est en plus antisymétrique, il s'agit d'une *relation d'ordre flou*.

D.1.4 Normes et conormes triangulaires

Les opérations ensemblistes floues classiques utilisent les fonctions min et max pour exprimer respectivement l'intersection et l'union de sous-ensembles flous. Il ne s'agit en fait que d'un cas particulier de norme triangulaire (ou t-norme) et de conorme triangulaire (ou t-conorme). D'une manière plus générale :

Définition 63 (t-norme) *Une t-norme est une fonction* $\top : [0, 1] \times [0, 1] \rightarrow [0, 1]$ *qui pour tout élément* $x, y, z, t \in [0, 1]$ *possède les propriétés suivantes :*
- *commutativité :* $\top(x, y) = \top(y, x)$,
- *associativité :* $\top(x, \top(y, z)) = \top(\top(x, y), z)$
- *monotonie :* $\top(x, y) \leq \top(z, t)$ *si* $x \leq z$ *et* $y \leq t$
- 1 *est élément neutre :* $\top(x, 1) = x$

La table D.1 donne quelques exemple de t-normes.

Définition 64 (t-conorme) *Une t-conorme est une fonction* $\perp : [0, 1] \times [0, 1] \rightarrow [0, 1]$ *qui pour tout élément* $x, y, z, t \in [0, 1]$ *possède les propriétés suivantes :*
- *commutativité :* $\perp(x, y) = \perp(y, x)$,
- *associativité :* $\perp(x, \perp(y, z)) = \perp(\perp(x, y), z)$
- *monotonie :* $\perp(x, y) \leq \perp(z, t)$ *si* $x \leq z$ *et* $y \leq t$
- 0 *est élément neutre :* $\perp(x, 0) = x$

La table D.2 donne quelques exemples de t-conormes.

Le choix des normes se fait en fonction du type d'opération à réaliser et du contexte. Une t-norme et une t-conorme sont dites duales pour l'opération de complémentation standard si elles satisfont aux lois de *De Morgan* :

$$1 - \perp(u, v) = \top(1 - u, 1 - v) \ \forall\, u, v \in [0, 1] \tag{D.11}$$

et :

$$1 - \top(u, v) = \perp(1 - u, 1 - v) \ \forall\, u, v \in [0, 1] \tag{D.12}$$

Binaire	Flou
E	μ
E^C	$1-\mu$
$\cap$	t-norme
$\cup$	t-conorme
$\forall$	inf
$\exists$	sup
Card	$\sum$

TABLE D.3 – Equivalences utilisées en extension formelle

D.1.5 Principe d'extension

Si la logique floue est une extension de la logique classique, toute relation ou opération définies sur des ensembles binaires doivent pouvoir être étendues aux ensembles flous. C'est à cet effet que ZADEH ([Zad75]) a introduit le principe d'extension :

Définition 65 (Principe d'extension de ZADEH ([Zad75])) *Considérons deux ensembles E et F, étant donné un sous-ensemble flou A de E et une application ϕ de E dans F, l'extension de ϕ à A est un sous-ensemble flou de F défini par*

$$\forall y \in F, \begin{cases} \mu_B(y) = \sup_{x \in E | y=\phi(x)} \mu_A(x) \textit{ si } \phi^{-1}(y) \neq \emptyset \\ \mu_B(y) = 0 \textit{ sinon.} \end{cases} \tag{D.13}$$

Ce principe est intéressant mais bien souvent lourd à mettre en pratique particulièrement pour les opérations ou relations définies sur plusieurs ensembles pour lesquelles il faut utiliser le supremum du produit cartésien. C'est pourquoi les extensions sont souvent réalisées de manière formelle en remplaçant les opérations binaires par leurs équivalents en logique flou (table D.3 extraite de [Tre98]).

D.2 Raisonnements en logique floue

D.2.1 Variable linguistique

En logique floue, les concepts manipulés sont représentés par des *variables linguistiques* ou *attributs linguistiques*. Une variable linguistique est déterminée par un triplet $\{X, E, T\}$ où X est une variable définie sur E et $T = \{A_1, A_2, ..., A_n\}$ un ensemble de sous-ensembles flous de E caractérisant X. A chaque élément de T est souvent associé un nom (voir figure D.3 page suivante) d'où la notion de variable *linguistique*.

Grâce a cette notion de variable linguistique, un ensemble de référence comprenant des informations vagues et imprécises peut se décomposer en plusieurs sous-ensembles flous parfaitement définis qui vont ainsi *granulariser* l'espace. Cette étape est appelé la *granularisation* de l'espace des connaissances qui est primordiale en logique floue.

D.2.2 Propositions et règles floues

Contrairement à la logique classique qui ne manipule que des propositions vraies ou fausses, les propositions floues sont caractérisées par un degré de vérité compris entre 0 et 1. Si $\{X, E, T\}$ est une variable linguistique, « X est A » par exemple constitue une proposition floue.

Une règle floue est définie comme une proposition floue correspondant à la mise en relation de deux propositions floues par une implication : « **Si** X est A **alors** Y est B ». L'implication utilisée pour relier

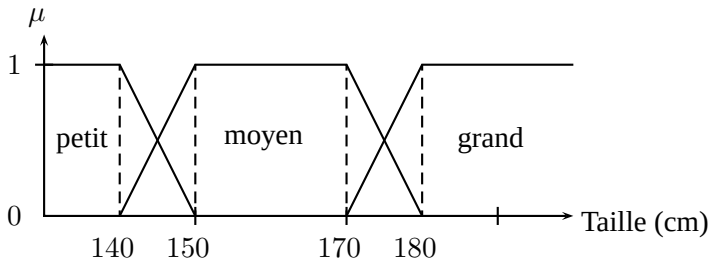


FIGURE D.3 – Variable linguistique (Taille, $\mathbb{R}^+$, {« grand », « moyen », « petit »})

$A \diagdown B$	0	1
0	1	1
1	0	1

TABLE D.4 – Implication classique

le prémisse et la conclusion d'une règle est une relation floue qui peut être définie de différentes manières suivant le contexte (la table D.5 donne différentes valeurs de vérité pour l'implication floue $\mathbf{I}(A, B)$).

Notons que certaines implications floues généralisent l'implication classique mais que d'autres, comme celles de Mamdani et Larsen (table D.5) par exemple, utilisent des opérateurs du type t-norme.

Les propositions floues peuvent être complexifiées comme en logique classique en utilisant des opérateurs logiques flous : la négation ($p_X =$ « X est A », non$(p_X) =$ « X n'est pas A »), la conjonction (« et » ou $\wedge$) et la disjonction (« ou » ou $\vee$). La négation utilise la plupart du temps l'opérateur de complémentation, les opérateurs de conjonction et disjonctions utilisent respectivement une t-norme et une t-conorme.

Nom	Degré de vérité de $\mathbf{I}(A, B)$
Reichenbach	$1 - \mu_A + \mu_A\mu_B$
Willmott	$max(1 - \mu_A, min(\mu_A, \mu_B))$
Rescher-Gaines	$\begin{cases} 1\ si\ \mu_A \leq \mu_B \\ 0\ sinon \end{cases}$
Kleene-Dienes	$max(1 - \mu_A, \mu_B)$
Brouwer-Gödel	$\begin{cases} 1\ si\ \mu_A \leq \mu_B \\ \mu_B\ sinon \end{cases}$
Goguen	$\begin{cases} min(\mu_B/\mu_A, 1)\ si\ \mu_A \neq 0 \\ 1\ sinon \end{cases}$
Lukasiewicz	$min(1 - \mu_A + \mu_B, 1)$
Mamdani	$min(\mu_A, \mu_B)$
Larsen	$\mu_A.\mu_B$

TABLE D.5 – Exemples d'implications floues

D.2.3 Raisonnement par déduction

Le *modus ponens* utilisé pour les raisonnements classiques peut être généralisé en logique floue :

Définition 66 Modus ponens généralisé *Etant donné une règle floue R : « Si X est A alors Y est B » et une proposition floue P : « X est A' ». Alors il existe un sous-ensemble flou B' décrivant Y défini par sa fonction d'appartenance de la façon suivante :*

$$\forall y,\ \mu_{B'}(y) = Sup_x \top(\mu_{A'}(x), \boldsymbol{I}(\mu_A(x), \mu_B(y))) \tag{D.14}$$

où $\top$ est une t-norme appelée ponens *et $\boldsymbol{I}$ une implication floue.*

Le choix de la fonction *ponens* et de l'implication est important puisqu'il assure ou non les raisonnements graduels du type « B' est d'autant plus proche de B que A' l'est de A ».

D.2.4 Systèmes d'inférence floue

Définition 67 (Système d'inférence flou) *Un système d'inférence flou est un système flou comprenant plusieurs règles.*

D.2.4.1 Une seule entrée

Considérons le cas d'un système à une entrée X et r règles $R_k,\ k = 1, ..., r$ qui portent toutes sur la même sortie Y. Les règles sont de la forme « R_k **:** si X est A alors Y est B_k ». Si on note $\mu_{A'}(X)$ le degré d'appartenance de X à A', il y a deux façons de trouver la fonction d'appartenance globale des sorties Y aux sous-ensembles flous B'.

La première approche est celle de *l'inférence locale* qui mixe les résultats de chaque règle appliquée séparemment des autres. Ainsi :

$$\begin{array}{c} \forall y,\ \mu_{B'}(y) = \top_{k=1,..,r}(\mu_{B'_k}(y)) \\ \Leftrightarrow\ \mu_{B'}(y) = \top_{k=1,..,r}\left(sup_x(Ponens(\mu_{A'}(x), \mathbf{I}(\mu_A(x), \mu_{B_k}(y))))\right) \end{array} \tag{D.15}$$

La deuxième approche consiste à construire une nouvelle règle R qui se substitue aux R_k. C'est la méthode d'*inférence globale* ou d'*agrégation*. Dans ce cas :

$$\forall y,\ \mu_{B'}(y) = sup_{x \in E_A}(Ponens(\mu_{A'}(x), \top_{k=1,..,r}(\mathbf{I}(\mu_A(x), \mu_{B_k}(y))))) \tag{D.16}$$

Remarquons que dans les deux approches, une T-norme est utilisée si l'implication est une généralisation du cas classique. Avec les implications de Mamdani et Larsen (table D.5 page précédente) il faudrait plutôt utiliser une T-conorme.

L'utilisation de l'une ou l'autre des solutions dépend du contexte. L'agrégation demande plus de calculs (l'opération sup est faite en dernier) mais l'inférence locale est souvent moins restrictive dans le cas des implications généralisant l'implication classique.

D.2.4.2 Plusieurs entrées non floues : cas de la classification

Prenons maintenant le cas plus général d'un système à plusieurs entrées ($X = \{X_i\},\ i \in [1, n]$) non floues comme étant par exemple les caractéristiques de forme d'un objet dans son espace de représentation. Les sorties sont les classes $\{\omega_j\},\ j \in [1, c]$. L'espace des entrées est découpé en sous-ensembles flous qui définissent l'ensemble des règles de classification : R_k : "Si X_1 est A_{k1}, ... , X_n est A_{kn} alors X appartient à ω_1 avec un degré b_{k1}, ... , X appartient à ω_c avec un degré b_{kc}."

Qui peuvent également être écrits, avec le formalisme flou, de la façon suivante en introduisant une variable de sortie Y :

R_k : "Si X_1 est A_{k1}, ... , X_n est A_{kn} alors Y_1 est B_{k1}, ... , Y_c est B_{kc}.", où les $B_{kj} = \{b_{kj}\}$ sont des singletons représentant l'appartenance à la classe ω_j.

Alors pour une donnée d'entrée X', son appartenance à chacune des classes est obtenue en utilisant une composition de type sup-min :

$$\mu_{\omega_j}(X') = \sup_i \min(\beta_i(X'), b_{ji}) \tag{D.17}$$

où $\beta_i(X')$ représente le degré de satisfaction de X' à la i$^{\text{e}}$ règle des R_k, par exemple :

$$\beta_i(X') = \min(\mu_{A_{1i}}, \ldots, \mu_{A_{ni}}) \tag{D.18}$$

BIBLIOGRAPHIE

[AKHM80] A. AMIN, A. KACED, J.P. HATON et R. MOHR : Handwritten arabic character recognition by the IRAC system. *In 5th International Conference on Pattern Recognition, Miami, USA*, pages 729–731, 1980.

[AM70] J.G. AUGUSTSON et J. MINKER : An analysis of some graph theoretical cluster techniques. *Journal of the ACM*, 17(4):571–588, 1970.

[Ami98] A. AMIN : Off-line arabic character-recognition : The state of the art. *Pattern Recognition*, 31(5):517–530, mai 1998.

[AMP84] Y.S. ABU-MOSTAFA et D. PSALTIS : Recognitive aspects of moment invariants. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 6(6):698–706, novembre 1984.

[ARB05a] D. ARRIVAULT, N. RICHARD et P. BOUYER : A fuzzy hierarchical attributed graph approach for handwritten hieroglyphs description. *In 11th International Conference on Computer Analysis of Images and Patterns, Versailles, France*, page 748, 2005.

[ARB05b] D. ARRIVAULT, N. RICHARD et P. BOUYER : A fuzzy hierarchical attributed graph approach for handwritten hieroglyphs description and matching. *In 8th International Conference on Document Analysis and Recognition, Séoul, Corée du Sud*, pages 898–903, août 2005.

[ARFM+04] D. ARRIVAULT, N. RICHARD, C. FERNANDEZ-MALOIGNE, et P. BOUYER : Collaboration entre approches statistique et structurelle pour la reconnaissance de caractères anciens. *In 8ème Colloque International Francophone sur l'Ecrit et le Document, La Rochelle, France*, pages 197–202, juin 2004.

[ARFM+05] D. ARRIVAULT, N. RICHARD, C. FERNANDEZ-MALOIGNE, et P. BOUYER : Collaboration between statistical and structural approaches for old handwritten characters recognition. *In* Lecture Notes on COMPUTER SCIENCE, éditeur : *5th IAPR International Workshop, Graph Based Representation in Pattern Recognition, Poitiers, France*, volume 3434, pages 291–300. Springer, mars 2005.

[Ars03] J. ARSAC : Implications philosophiques de la science contemporaine. Rapport technique 3, Académie des Sciences Morales et Politiques, novembre 2003.

[AS66] M.S. ALI et S.D. SILVEY : A general class of coefficients of divergence of one distrubution from another. *Journal of Royal Statistical Society*, 28:131–142, 1966.

[Att95] D. ATTALI : *Squelettes et graphes de Voronoï 2D et 3D*. Thèse de doctorat, Université Joseph Fourier - Grenoble I, octobre 1995.

[AYV01] N. ARICA et F.T. YARMAN-VURAL : An overview of character recognition focused on off-line handwriting. *IEEE Trans. Systems, Man and Cybernetics*, 31(2):216–233, mai 2001.

[Bas88] M. BASSEVILLE : Distance measures for signal processing and pattern recognition. Rapport technique, INRIA - Rennes, septembre 1988.

[BAS96] S.O. BELKASIM, M. AHMADI et M. SHRIDHAR : Efficient algorithm for fast computation of zernike moments. *Journal of the Franklin Institute*, 333:577–581, juillet 1996.

[BB76] H.G. BARROW et R.M. BURSTALL : Subgraph isomorphism, matching relational structures and maximal cliques. *Information Processing Letters*, 4:83–84, 1976.

[BB98] J.P. BRAQUELAIRE et L. BRUN : Image segmentation with topological maps and interpixel representation. *Journal of Visual Communication and Image Representation*, 9:62–79, 1998.

[BBOM05] F. BORTOLOZZI, Jr.A. BRITTO, L.S. OLIVEIRA et M. MORITA : *Document Analysis*, chapitre Recent Advances in Handwriting Recognition, pages 1–31. ISBN 8177647849. Umapada Pal et al, 2005.

[BBPP99] I. BOMZE, M. BUDINICH, P. PARDALOS et M. PELILLO : *The maximum clique problem*, volume 4, pages 1–74. Du, D.-Z. and Pardalos, P.M., 1999.

[Bel01] A. BELAID : La reconnaissance automatique de l'écriture. *Dossier Pour la Science*, 33, octobre 2001.

[BGH+88] J. BUURMAN, N. GRIMAL, M. HAINSWORTH, J. HALLOF et D.V.D. PLAS : Inventaire des signes hiéroglyphiques en vue de leur saisie informatique. *In Mémoires de l'Académie des Inscripitions et Belles Lettres*, volume VIII. Institut de France, Paris, 1988.

[BH84] A. BELAID et J.P. HATON : A syntactic approach for handwritten mathematical formula recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 6(1):105–111, janvier 1984.

[BIK+02] Hervé BRÖNNIMANN, John IACONO, Jyrki KATAJAINEN, Pat MORIN, Jason MORRISON et Godfried TOUSSAINT : In-place planar convex hull algorithms. *In Latin American Theoretical INformatics, LATIN2002, Cancun, Mexico*, pages 494–507, 2002.

[Blo99] I. BLOCH : Fuzzy relative position between objects in image processing : a morphological approach. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 21(7):657–664, juillet 1999.

[Blu67] H. BLUM : A transformation for extracting new descriptions of shape. *In Models for the Perception of Speech and Visual Form*, pages 362–380. MIT Press, 1967.

[BM93] B. BOUCHON-MEUNIER : *La Logique Floue*. Presses Universitaires de France, 1993.

[BM99a] T.M. BERNARD et A. MANZANERA : Improved low complexity fully parallel thinning algorithm. *In 10th International Conference on Image Analysis and Processing, Venise, Italie*, page 215, Washington, DC, USA, 1999. IEEE Computer Society.

[BM99b] W. BOEHM et A. MÜLLER : On de casteljau's algorithm. *Computer Aided Geometric Design*, 16(7):587–605, 1999.

[BMRB96] B. BOUCHON-MEUNIER, M. RIFQI et S. BOTHOREL : Toward general measures of comparison of objects. *Fuzzy Sets and Systems*, 84:143–153, 1996.

[Bom97] M.I. BOMZE : Evolution towards the maximum clique. *J. of Global Optimization*, 10(2): 143–164, 1997.

[Bre96] L. BREIMAN : Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[Bri99] E. BRIBIESCA : A new chaine code. *Pattern Recognition*, 2(32):235–251, février 1999.

[Bru96] L. BRUN : *Segmentation d'images à base topologique*. Thèse de doctorat, Universitée de Bordeaux I, 1996.

[BS97] A. BELAID et G. SAON : Utilisation des processus markoviens en reconnaissance de l'écriture. *revue Traitement du Signal*, 14(2):161–177, 1997.

[BSA91] S.O. BELKASIM, M. SHRIDHAR et M. AHMADI : Pattern recognition with moment invariants : A comparative study and new results. *Pattern Recognition*, 24(12):1117–1138, 1991.

[BSA92] S. BELKASIM, M. SHRIDHAR et M. AHMADI : Pattern classification using an efficient KNNR. *Pattern Recognition*, 25(10):1269–1274, 1992.

[Bun97] H. BUNKE : On a relation between graph edit distance and maximum common subgraph. *Pattern Recognition Letters*, 18(8):689–694, août 1997.

[Bun00] H. BUNKE : Recent developments in graph matching. *In 15th International Conference on Pattern Recognition, Barcelona, Espagne*, pages Vol II : 117–124, 2000.

[Bun01] H. BUNKE : Recent advances in structural pattern recognition with applications to visual form analysis. *In Workshop on Visual Form*, page 11 ff., 2001.

[Bun03] H. BUNKE : Recognition of cursive roman handwriting past, present and future. *In 7th International Conference on Document Analysis and Recognition, Edinburgh, Ecosse*, pages 448–459, 2003.

[CC92] K.P. CHAN et Y.S. CHEUNG : Fuzzy-attribute graph with application to chinese character recognition. *IEEE Trans. Systems, Man and Cybernetics*, 22(2):402–410, 1992.

[CG95] G. CELEUX et G. GOVAERT : Gaussian parsimonious clustering models. *Pattern Recognition*, 28(5):781–793, mai 1995.

[CH67] T.M. COVER et P.E. HART : Nearest neighbor pattern classification. *IEEE Trans. on Information Theory*, 13(1):21–27, 1967.

[Cha22] J.F. CHAMPOLLION : Lettre à M. Dacier. Chez Firmin Didot père et fils, imprimeur du Roi et de l'Institut, Paris, 1822. source : bibliothèque numérique Gallica.

[Cho56] N. CHOMSKY : Three models for the description of language. *IRE Transactions on Information Theory*, 2:113–124, 1956.

[CHS94] E. COHEN, J.J. HULL et S.N. SRIHARI : Control-structure for interpreting handwritten addresses. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 16(10):1049–1055, octobre 1994.

[CKP95] W.J. CHRISTMAS, J.V. KITTLER et M. PETROU : Structural matching in computer vision using probabilistic relaxation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 17(8):749–764, août 1995.

[CL96] R.G. CASEY et E. LECOLINET : A survey of methods and strategies in character segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(7):690–706, juillet 1996.

[CM98] M. COLLIER et B. MANLEY : *How to read egyptian hieroglyphs : a step-by-step guide to teach yourself*. The British Museum Press, 1998.

[CMK02] A. CORNUÉJOLS, L. MICLET et Y. KODRATOFF : *Apprentissage artificiel : concepts et algorithmes*. Eyrolle, 2002.

[CR92] L.J. CHIPMAN et H.S. RANGANATH : A fuzzy relaxation algorithm for matching imperfectly segmented images to models. *In IEEE Southeastcon'92*, volume 1, pages 128–136, avril 1992.

[CRM03] C.W. CHONG, P. RAVEENDRAN et R. MUKUNDAN : A comparative analysis of algorithms for fast computation of zernike moments. *Pattern Recognition*, 36(3):731–742, mars 2003.

[Csi63] I. CSISZÁR : Eine Informationtheoretische Ungleichung und ihre Anwendungen auf den Beweis der ergodizität von Markoffschen Ketten. *Magyar Tudományos Akadémia Közleményei*, 8:85–108, 1963.

[Dan98] V.M. DANG : *Classification de données spatiales : modèles probabilistes et critères de partitionnement*. Thèse de doctorat, Université Technologique de Compiègne, 1998.

[DBF04] G. DAMIAND, Y. BERTRAND et C. FIORIO : Topological model for two-dimensional image representation : Definition and optimal extraction algorithm. *Computer Vision and Image Understanding*, 93(2):111–154, février 2004.

[DE64] M. R. DAVIS et T. O. ELLIS : The rand tablet : A man-machine graphical communication device. *In Joint Computer Conference*, pages 325–333, 1964.

[Deu72] E.S. DEUTSCH : Thinning algorithms on rectangular, hexagonal, and triangular arrays. *Communications of the ACM*, 15(9):827–837, septembre 1972.

[Din97] X.Q. DING : *Handbook of Character Recognition and Document Image Analysis*, chapitre Machine printed chinese character recognition, pages 305–329. World Scientific, 1997.

[DJ94] M.P. DUBUISSON et A.K. JAIN : A modified hausdorff distance for object matching. *In 12th IAPR International Conference on Pattern Recognition, La Hague, Pays-Bas*, pages A :566–568, 1994.

[dlH05] C. de la HIGUERA : A bibliographical study of grammatical inference. *Pattern Recognition*, 38(9):1332–1348, septembre 2005.

[DM93] B. DUBUISSON et M. MASSON : A statistical decision rule with incomplete knowledge about classes. *Pattern Recognition*, 26(1):155–165, janvier 1993.

[Dom03] J. DOMBRE : *Systèmes de représentation multi-échelles pour l'indexation et la restauration d'archives médiévales couleurs.* Thèse de doctorat, Université de Poitiers, décembre 2003.

[Dub01] B. DUBUISSON : *Diagnostic, Intelligence Artificielle et Reconnaissance des Formes*, chapitre Diagnostic et reconnaissance des formes, pages 107–140. Hermès Science, 2001.

[EF84] M.A. ESHERA et K.S. FU : A graph distance measure for image analysis. *IEEE Trans. Systems, Man and Cybernetics*, 14(3):398–408, mai 1984.

[Eve99] M. EVERSON : Encoding egyptian hieroglyphs in plane 1 of the ucs. Rapport technique, International Organisation for Standardisation, janvier 1999. http ://std.dkuug.dk/jtc1/sc2/wg2/docs/n1944.pdf.

[Flu00] J. FLUSSER : On the independence of rotation moment invariants. *Pattern Recognition*, 33(9):1405–1410, septembre 2000.

[Fre61] H. FREEMAN : On the encoding of arbitrary geometric configuration. *IEEE Trans. Computer*, 2(10):260–268, juin 1961.

[FS03] J. FLUSSER et T. SUK : Construction of complete and independent systems of rotation moment invariants. *In 10th International Conference on Computer Analysis of Images and Patterns, Groningen, Pays-Bas*, pages 41–48, 2003.

[Fu82] K.S. FU : *Syntactic pattern recognition and applications*. Prentice Hall, 1982.

[FVBB01] M. FLIGNER, J. VERDUCCI, J. BJORAKER et P. BLOWER : A new association coefficient for molecular dissimilarity. *In The Second Joint Sheffield Conference on Chemoinformatics*, 2001.

[GAA+01] N. GORSKI, V. ANISIMOV, E. AUGUSTIN, O. BARET et S. MAXIMOV : Industrial bank check processing : the A2iA CheckReaderTM. *International Journal on Document Analysis and Recognition*, 3(4):196–206, 2001.

[Gac97] L. GACÔGNE : *Eléments de logique floue*. Hermes, 1997.

[GGD+98] E. GÓMEZ-SÁNCHEZ, J.A. GAGO-GONZÁLEZA, Y.A. DIMITRIADISA, J.M. CANO-IZQUIERDOB et J. LÓPEZ-CORONADO : Experimental study of a novel neuro-fuzzy system for on-line handwritten unipen digit recognition. *Pattern Recognition Letters*, 19(3-4):357–364, 1998.

[GR96] S. GOLD et A. RANGARAJAN : A graduated assignment algorithm for graph matching. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(4):377–388, avril 1996.

[Gra72] G. H. GRANLUND : Fourier Preprocessing for Hand Print Character Recognition. *IEEE Trans. on Computers*, C–21(2):195–201, 1972.

[Gro95] P.J. GROTHER : Handprinted forms and character database, nist special database 19. *In National Institute of Standards and Technology (NIST) Intelligent Systems Division*, 1995.

[GT78] R.C. GONZALEZ et M.G. THOMASON : *Syntactic Pattern Recognition : An Introduction*. Addison-Wesley, 1978.

[GW92] R.C. GONZALEZ et R.E. WOODS : *Digital Image Processing*. Addison Wesley, 1992.

[Har72] L.D. HARMON : Automatic recognition of print and script. *Proceedings of the IEEE*, 60(10):1165–1177, octobre 1972.

[Heb49] D.O HEBB : *The Organization of Behavior : A Neuropsychological Theory*. Wiley, 1949.

[Hil69] C.J. HILDITCH : Linear skeletons from square cupboards. *Machine Intelligence*, 4:403–420, 1969.

[HL93] T.H. HILDEBRANDT et W. LIU : Optical recognition of handwritten chinese characters : Advances since 1980. *Pattern Recognition*, 26(2):205–225, février 1993.

[HMMG01] A. HERO, B. MA, O. MICHEL et J. GORMAN : Alpha-divergence for classification, indexing and retrieval. Rapport technique CSPL-328, Communications and Signal Processing Laboratory, The University of Michigan, mai 2001.

[Hot33] H. HOTELLING : Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441 and 498–520, 1933.

[Hu62] M.K. HU : Visual pattern recognition by moment invariants. *IEEE Trans. on Information Theory*, 8(2):179–187, février 1962.

[Jai89] A.K. JAIN : Fundamentals of digital image processing. *In Prentice Hall*, 1989.

[JDM00] A.K. JAIN, R.P.W. DUIN et J. MAO : Statistical pattern recognition : A review. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(1):4–37, janvier 2000.

[JMB01] X. JIANG, A. MUNGER et H. BUNKE : On median graphs : Properties, algorithms, and applications. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(10):1144–1151, octobre 2001.

[Kar72] R.M. KARP : Reducibility among combinatorial problems. *In* R.E. MILLER et J.W. THATCHER, éditeurs : *Complexity of computer computations*, pages 85–103, 1972.

[KB00] G. KAUFMANN et H. BUNKE : Automated reading of cheque amounts. *Pattern Analysis & Applications*, 3(2):132–141, 2000.

[KH90a] A. KHOTANZAD et Y.H. HONG : Invariant image recognition by zernike moments. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 12(5):489–497, mai 1990.

[KH90b] A. KHOTANZAD et Y.H. HONG : Rotation invariant image recognition using features selected via a systematic method. *Pattern Recognition*, 23(10):1089–1101, 1990.

[KHP05] G. KOCH, L. HEUTTE et T. PAQUET : Automatic extraction of numerical sequences in handwritten incoming mail documents. *Pattern Recognition Letters*, 26(8):1118–1127, juin 2005.

[KK00] H.K. KIM et J.D. KIM : Region-based shape descriptor invariant to rotation, scale and translation. *Signal Processing : Image Communication*, 16(1-2):87–93, septembre 2000.

[KK01] H.Y. KIM et J.H. KIM : Hierarchical random graph representation of handwritten characters and its application to hangul recognition. *Pattern Recognition*, 34(2):187–201, février 2001.

[KKS00] J.H. KIM, K.K. KIM et C.Y. SUEN : Hybrid schemes of homogeneous and heterogeneous classifiers for cursive word recognition. *In 10th International Workshop on Frontiers in Handwriting Recognition, Amsterdam, Netherlands*, pages 433–442, 2000.

[KL63] L.A. KAMENTSKY et C.N. LIU : Computer-automated design of multifont print recognition. *IBM Journal of Research and Development*, 7(1):2–13, 1963.

[KL97] J. KANAI et Y. LIU : *Handbook of Character Recognition and Document Image Analysis*, chapitre An OCR-oriented overview of ideographic writing systems, pages 285–304. World Scientific, 1997.

[KLSS02] A.L. KOERICH, Y. LEYDIER, R. SABOURIN et C.Y. SUEN : A hybrid large vocabulary handwritten word recognition system using neural networks with hidden markov models. *In 11th International Workshop on Frontiers in Handwriting Recognition, Ontario, Canada*, pages 99–104, 2002.

[KO90] H. KALVIAINEN et E. OJA : Comparisons of attributed graph matching algorithms for computer vision. *In Finnish Artificial Intelligence Symposium, Oulu, Finlande*, pages 354–368, 1990.

[LBBH98] Y. LECUN, L. BOTTOU, Y. BENGIO et P. HAFFNER : Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, novembre 1998.

[Leb05] G. LEBRUN : Utilisation des pyramides combinatoires pour la détection de manuscrits hiéroglyphiques. Mémoire de D.E.A., ENSICAEN, 2005.

[LG02] J. LIU et P. GADER : Neural networks with enhanced outlier rejection ability for off-line handwritten word recognition. *Pattern Recognition*, 35(10):2061–2071, octobre 2002.

[Liu64] C.N. LIU : A programmed algorithm for designing multifont character recognition logic. *IEEE Trans. on Electronic Computers*, EC-13:586–594, octobre 1964.

[LKFK02] N. LIOLIOS, E. KAVALLIERATOU, N. FAKOTAKIS et G. KOKKINAKIS : A new shape transformation approach to handwritten character recognition. *In 16th IAPR International Conference on Pattern Recognition, Quebec City, Canada*, pages I : 584–587, 2002.

[LLG95] E. LETHELIER, M. LEROUX et M. GILLOUX : An automatic reading system for handwritten numeral amounts on french checks. *In 3th International Conference on Document Analysis and Recognition, Montreal, Canada*, pages I :92–97, 1995.

[LLS92] L. LAM, S.W. LEE et C.Y. SUEN : Thinning methodologies - a comprehensive survey. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 14(9):869–885, septembre 1992.

[Lon98] S. LONCARIC : A survey of shape analysis techniques. *Pattern Recognition*, 31(8):983–1001, août 1998.

[MA04] A.R. MUGDADI et I.A. AHMAD : A bandwidth selection for kernel density estimation of functions of random variables. *Computational Statistics & Data Analysis*, 47:49–62, 2004.

[Mal96] A. MALAVIYA : *On-line Handwriting Recognition with a Fuzzy Feature Description Language*. Thèse de doctorat, Technische Universität Berlin, 1996.

[Man86] J. MANTAS : An overview of character recognition methodologies. *Pattern Recognition*, 19(6):425–430, 1986.

[MK97] S. MADHVANATH et V. KRPASUNDAR : Pruning large lexicons using generalized word shape descriptors. *In 4th International Conference on Document Analysis and Recognition, Ulm, Allemagne*, page Poste, 1997.

[MO98] L. MICO et J. ONCINA : Comparison of fast nearest neighbor classifiers for handwritten character recognition. *Pattern Recognition Letters*, 19(3-4):351–356, mars 1998.

[MP93] G.T. MAN et J.C. POON : A fuzzy-attributed graph approach to handwritten character recognition. *In Second IEEE Int. Conf. on Fuzzy Systems*, pages 570–575, San Francisco, mars 1993.

[MP97] A. MALAVIYA et L. PETERS : Fuzzy feature description of handwriting patterns. *Pattern Recognition*, 30(10):1591–1604, octobre 1997.

[MP00] A. MALAVIYA et L. PETERS : Fuzzy handwriting description language : Fohdel. *Pattern Recognition*, 33(1):119–131, janvier 2000.

[MPEA05] V. MÄRGNER, M. PECHWITZ et H. EL ABED : ICDAR 2005 arabic handwriting recognition competition. *In 8th International Conference on Document Analysis and Recognition, Séoul, Corée du Sud*, pages 70–74, 2005.

[MR95] R. MUKUNDAN et K.R. RAMAKRISHNAN : Fast computation of legendre and zernike moments. *PR*, 28(9):1433–1442, septembre 1995.

[MYY84] S. MORI, K. YAMAMOTO et M. YASUDA : Research on machine recognition of handprinted characters. *PAMI*, 6(4):386–405, juillet 1984.

[NC66] G. NAGY et R.G. CASEY : Recognition of printed chinese characters. *IEEE Trans. on Electronic Computers*, 15(1):91–101, février 1966.

[NS66] G. NAGY et J.R. SHELTON : Self-corrective character recognition system. *IEEE Trans. on Information Theory*, 12(2):215–222, avril 1966.

[Par62] E. PARZEN : On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076, septembre 1962.

[Pav78] T. PAVLIDIS : A review of algorithms for shape analysis. *Computer Graphics Image Processing*, 7(2):243–258, avril 1978.

[Pav82] T. PAVLIDIS : Algorithms for graphics and image processing. *In Computer Science Press*, 1982.

[PB02] A. PERCHANT et I. BLOCH : Fuzzy morphisms between graphs. *Fuzzy Sets and Systems*, 128:149–168, 2002.

[PC99] J.S. PARK et D.H. CHANG : 2-D invariant descriptors for shape-based image retrieval. *In 26th KISS Spring Conference*, pages 465–474, Kwangwoon University, Corée, 1999.

[PCB94] A.R. PEARCE, T.M. CAELLI et W.F. BISCHOF : Rulegraphs for graph matching in pattern recognition. *Pattern Recognition*, 27(9):1231–1247, septembre 1994.

[PDM86] S.K. PAL et D.K. DUTTA-MAJUMDER : *Fuzzy mathematical approach to pattern recognition*. Halsted Press, New York, NY, USA, 1986.

[Pel99] M. PELILLO : Replicator equations, maximal cliques, and graph isomorphism. *Neural Computation*, 11(8):1933 – 1955, novembre 1999.

[PF77] E. PERSOON et K.S. FU : Shape discrimination using Fourier descriptors. *IEEE Trans. Systems, Man and Cybernetics*, SMC-7(3), mars 1977.

[PI97] M. PEURA et J. IIVARINEN : Efficiency of simple shape descriptors. *In Visual Form : Analysis and Recognition*, pages 443–451, 1997.

[PI98] S. PROCTER et J. ILLINGWORTH : Combining HMM classifiers in a handwritten text recognition system. *In IEEE International Conference on Image Processing*, Chicago, USA, octobre 1998.

[Pot64] R.J. POTTER : An optical character scanner. *J. Soc. Photographic Instrumentation Engineers*, 2:75–78, février et mars 1964.

[Pra01] W.K. PRATT : *Digital Image Processing (Third Edition)*. ISBN 0-471-37407-5. Wiley, 2001.

[PSBB93] R. PLAMONDON, C.Y. SUEN, M. BOURDEAU et C. BARRIERE : Methodologies for evaluating thinning algorithms for character recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 7:1247–1270, 1993.

[PSZ98] M. PELILLO, K. SIDDIQI et S.W. ZUCKER : Matching hierarchical structures using association graphs. *In 5th European Conference on Computer Vision, Freiburg, Allemagne*, page II : 3, 1998.

[Ram72] U. RAMER : An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics Image Processing*, 1:244–256, 1972.

[RC92] H.S. RANGANATH et L.J. CHIPMAN : Fuzzy relaxation approach for inexact scene matching. *Image and Vision Computing*, 10(9):631–640, 1992.

[Rei69] G.M. REICHER : Perceptual recognition as a function of meaningfulness of stimulus material. *Journal of Experimental Psychology*, 81:275–280, 1969.

[RF03] A.F.R. RAHMAN et M.C. FAIRHURST : Multiple classifier decision combination strategies for character recognition : A review. *International Journal on Document Analysis and Recognition*, 5(4):166–194, juillet 2003.

[Ris01] I. RISH : An empirical study of the naive bayes classifier. *In IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence*, pages 41–46, 2001.

[RM96] S. RANGARAJAN, A. et Gold et E. MJOLSNESS : A novel optimizing network architecture with applications. *Neural Computation*, 8(5):1041–1060, juillet 1996.

[Ron88] Christian RONSE : Minimal test patterns for connectivity preservation in parallel thinning algorithms for binary digital images. *Discrete Applied Mathematics*, 21(1):67–79, 1988.

[Ros74] A. ROSENFELD : Adjacency in digital pictures. *In Information and Control*, 26, 1974.

[Ros99] P.L. ROSIN : Measuring rectangularity. *Machine Vision and Applications*, 11(4):191–196, 1999.

[Ros03] P.L. ROSIN : Measuring shape : Ellipticity, rectangularity, and triangularity. *Machine Vision and Applications*, 14(3):172–184, juillet 2003.

[RP66] A. ROSENFELD et J.L. PFALTZ : Sequential operations in digital picture processing. *Journal of the ACM*, 13(4):471–494, octobre 1966.

[RP00] V. RAMASUBRAMANIAN et K.K. PALIWAL : Fast nearest-neighbor search algorithms based on approximation-elimination search. *Pattern Recognition*, 33(9):1497–1510, septembre 2000.

[RSG92] T.W. RAUBER et A.S. STEIGER-GARÇÃO : Shape description by UNL Fourier features-an application to handwritten character recognition. *In 11th IAPR International Conference on Pattern Recognition, Jerusalem, Israël*, 1992.

[Ruc96] W.J. RUCKLIDGE : *Efficient Visual Recognition Using the Hausdorff Distance*. ISBN 3540619933. Springer-Verlag New York, Inc., 1996.

[Rut66] D. RUTOVITZ : Pattern recognition. *Journal of Royal Statistical Society*, 129:504–530, 1966.

[SA99] S. SINGH et A. AMIN : Fuzzy recognition of chinese characters. *In Irish Machine Vision and Image Processing Conference (IMVIP'99)*, pages 219–227, Dublin, 1999.

[Say73] K.M. SAYRE : Machine recognition of handwritten words : A project report. *Pattern Recognition*, 5(3):213–228, septembre 1973.

[Sch90] R.E. SCHAPIRE : The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.

[Sch02] R.E. SCHAPIRE : The boosting approach to machine learning : An overview. *In MSRI Workshop on Nonlinear Estimation and Classification*, pages 149–172, 2002.

[Shu00] L. SHU : An inference method for fuzzy tree grammars. *Fuzzy Sets Syst.*, 112(1):173–176, 2000.

[Sim84] J.C. SIMON : *La Reconnaissance des Formes par Algorithmes*. Masson, 1984.

[Sin64] R. SINKHORN : A relationship between arbitrary positive matrices and doubly stochastic matrices. *Annals of Mathematical Statistics*, 35:876–879, 1964.

[SJ01] C. SAINT-JEAN : *Classification paramétrique robuste partiellement supervisée en reconnaissance des formes*. Thèse de doctorat, Université de La Rochelle, 2001.

[SKK04] T.B. SEBASTIAN, P.N. KLEIN et B.B. KIMIA : Recognition of shapes by editing their shock graphs. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 26(5):550–571, mai 2004.

[SKP99] D.G. SIM, O.K. KWON et R.H. PARK : Object matching algorithms using robust hausdorff distance measures. *IEEE Trans. on Image Processing*, 8(3):425–429, mars 1999.

[SKS95] K. SIDDIQI, B.B. KIMIA et C.W. SHU : Geometric shock capturing ENO schemes for subpixel interpolation, computation and curve evolution. *In Symposium on Computer Vision,Miami Beach, FL, USA*, pages 437–442, 1995.

[SM93] M. SCHMITT et J. MATTIOLI : *Morphologie Mathématique*. Logique Mathématiques Informatique. Masson, Paris, 1993.

[SMle02] R. SADYKHOV, O. MALENKO, A.N. LIMOVICH et M.L. ELINGER : Hybrid system for recognition of handwritten symbols on the base of structural methods and neural networks. *In Vision Interface Conference - Minsk - Belarussie*, page 223, 2002.

[SR71] R. STEFANELLI et A. ROSENFELD : Some parallel thinning algorithms for digital pictures. *Journal of the ACM*, 18(2):255–264, avril 1971.

[SSDZ99] K. SIDDIQI, A. SHOKOUFANDEH, S.J. DICKINSON et S.W. ZUCKER : Shock graphs and shape matching. *International Journal of Computer Vision*, 35(1):13–32, novembre 1999.

[Sta72] W.W. STALLINGS : Recognition of printed chinese characters by automatic pattern analysis. *CGIP*, 1(1):47–65, 1972.

[Ste74] M. A. STEPHENS : EDF statistics for goodness of fit and some comparisons. *Journal of the American Statistical Association*, 69:730–737, 1974.

[Sue86] C.Y. SUEN : Character recognition by computer and applications. *In Handbook of Pattern Recognition and Image Processing*, pages 569–586, 1986.

[Tam78] H. TAMURA : A comparison of line thinning algorithms from digital geometry viewpoint. *In 4th IAPR International Conference on Pattern Recognition, Kyoto, Japon*, pages 715–719, 1978.

[TC88] C.H. TEH et R.T. CHIN : On image analysis by the methods of moments. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 10(4):496–513, juillet 1988.

[TCL+98] Y.Y. TANG, M. CHERIET, J. LIU, J.N. SAID et C.Y. SUEN : *Handbook of Pattern Recognition and Computer Vision*, chapitre Chapter 8 : Document Analysis and Recognition by Computers. World Scientific Publishing Company, 1998.

[Tea80] M.R. TEAGUE : Image analysis via the general theory of moments. *Journal of the Optical Society of America*, 70(8):920–930, août 1980.

[TF79] W.H. TSAI et K.S. FU : Error-correcting isomorphisms of attributed relational graphs for pattern analysis. *IEEE Trans. Systems, Man and Cybernetics*, 9(12):757–768, décembre 1979.

[TF83] W.H. TSAI et K.S. FU : Subgraph error-correcting isomorphisms for syntatic pattern recognition. *IEEE Trans. Systems, Man and Cybernetics*, 13(1):48–62, janvier 1983.

[TJT96] O.D. TRIER, A.K. JAIN et T. TAXT : Feature-extraction methods for character-recognition : A survey. *Pattern Recognition*, 29(4):641–662, avril 1996.

[TK03] H. TEK et B.B. KIMIA : Symmetry maps of free-form curve segments via wave propagation. *International Journal of Computer Vision*, 54(1-3):35–81, août 2003.

[Tre98] B. TREMBLAIS : Mise en correspondance de graphes d'attributs flous, application à la reconnaissance de structures cérébrales dans les images irm à partir d'un atlas anatomique. Mémoire de D.E.A., ENST Paris, septembre 1998.

[TSW90] C.C. TAPPERT, C.Y. SUEN et T. WAKAHARA : The state of the art in on-line handwriting recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 12(8):787–808, août 1990.

[Tub89] J.D. TUBBS : A note on binary template matching. *Pattern Recognition*, 22(4):359–365, 1989.

[Ull76] J.R. ULLMANN : An algorithm for subgraph isomorphism. *Journal of the ACM*, 23(1):31–42, janvier 1976.

[Vin02] A. VINCIARELLI : A survey on off-line cursive word recognition. *Pattern Recognition*, 35(7):1433–1446, juillet 2002.

[Won87] A.K.C. WONG : Structural pattern recognition : A random graph approach. *Pattern Recognition Theory and Application*, 87:323–345, 1987.

[WY85] A.K.C. WONG et M. YOU : Entropy and distance of random graphs with application to structural pattern recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 7(5):599–609, septembre 1985.

[YSB89] B. YANG, W.E. SNYDER et G.L. BILBRO : Matching oversegmented 3D images to models using association graphs. *Image and Vision Computing*, 7(2):135–143, mai 1989.

[YTF73] Y. YOKOI, J.I. TORIWAKI et T. FUKUMURA : Topological properties in digitized binary pictures. *Systems Computers Controls*, 4(6):32–39, 1973.

[YTF75] Y. YOKOI, J.I. TORIWAKI et T. FUKUMURA : An analysis of topological properties of digitized binary pictures using local features. *Journal of Computer Graphics and Image Processing*, 4:63–73, 1975.

[Zad65] L.A. ZADEH : Fuzzy sets. *InfoControl*, 8:338–353, 1965.

[Zad71] Lotfi A. ZADEH : Similarity relations and fuzzy orderings. *Information Sciences*, 3:177–200, 1971.

[Zad75] L.A. ZADEH : The concept of a linguistic variable and its application to approximate reasoning, I, II and III. *Information Sciences*, 8, 1975.

[Zad00] L.A. ZADEH : From computing with numbers to computing with words - from manipulation of measurements to manipulation of perceptions. *In Intelligent Systems and Soft Computing : Prospects, Tools and Applications*, pages 3–40, London, UK, 2000. Springer-Verlag.

[ZC06] S.K. ZHOU et R. CHELLAPPA : From sample similarity to ensemble similarity : Probabilistic distance measures in reproducing kernel hilbert space. *IEEE Trans. Pattern Analysis and Machine Intelligence*, à paraître, 2006.

[ZCSY03] S. ZHAO, Z. CHI, P. SHI et H. YAN : Two-stage segmentation of unconstrained handwritten chinese character. *Pattern Recognition*, 36(1):145–156, 2003.

[Zer34] F. ZERNIKE : Diffraction theory of the cut procedure and its improved form, the phase contrast method. *Physica*, 1:689–704, 1934.

[ZL04] D. ZHANG et G. LU : Study and evaluation of different Fourier methods for image retrieval. *Image and Vision Computing*, 23(1):33–49, janvier 2004.

[ZR72] C. T. ZAHN et R. Z. ROSKIES : Fourier descriptors for plane closed curves. *IEEE Trans. Computers*, C-21:269–281, 1972.

[ZS84] T.Y. ZHANG et C.Y. SUEN : A fast parallel algorithm for thinning digital patterns. *Communications of the ACM*, 27(3):236–240, mars 1984.

[ZS04] B. ZHANG et S.N. SRIHARI : Fast k-nearest neighbor classification using cluster-based trees. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 26(4):525–528, avril 2004.

[ZW94] Y.Y. ZHANG et P.S.P. WANG : A new parallel thinning methodology. *International Journal of Pattern Recognition and Artificial Intelligence*, 8:999–1011, 1994.

Apport des Graphes dans la Reconnaissance Non-Contrainte de Caractères Manuscrits Anciens

Résumé L'objectif des travaux réalisés au cours de cette thèse est d'adresser la problématique de la reconnaissance générique de caractères manuscrits par les méthodes structurelles à base de graphes. Les écrits traités sont non-contraints et hétérogènes dans le temps. Les méthodes classiques, dites statistiques, sont efficaces mais ne peuvent s'appliquer qu'à des écritures à vocabulaire restreint dans le cadre d'un système avec une phase d'apprentissage. Nous proposons deux systèmes de reconnaissance à base de graphes d'attributs. Le premier utilise des attributs numériques et une modélisation de la base d'apprentissage avec des graphes aléatoires. L'intégration des informations de structure change la notion de complexité et permet une coopération intéressante avec les approches statistiques. Le second système utilise des attributs hiérarchiques flous. Il permet une reconnaissance sans apprentissage basée sur des modèles qui tend vers la reconnaissance générique recherchée.

Graphs Contribution in Unconstraint Old Handwritten Characters Recognition

Abstract Our work has been motivated by the issue of generic handwritten characters recognition. We try to address it with structural methods based on graph modelling. The documents processed are unconstrained and come from different periods. Classical statistical methods are efficients but they can only process languages with restrained vocabulary according to a learning phase. Two recognition systems based on attributed graphs are proposed. The first one uses numerical attributes and random graphs for modelling the learning base. The structural information changes the complexity notion and allows an interesting cooperation with statistical methods. The second one uses hierarchical fuzzy attributes. It is a model-based recognition system with no learning phase. It brings an interesting first step for generic recognition.

Discipline : traitement du signal et des images.

Mots clés : reconnaissance de caractères manuscrits, reconnaissance structurelle, graphe d'attributs, graphe aléatoires, graphe d'attributs hiérarchiques flous, caractères anciens, écritures non-contraintes.

Denis Arrivault
Laboratoire Signal Image Communication
Bât. SP2MI - Téléport 2, Boulevard Marie et Pierre Curie
BP 30179 86962 Futuroscope Chasseneuil Cedex