وزارة التعليم العالي و البحث العلمي
BADJI MOKHTAR – ANNABA UNIVERSITY
UNIVERSITE BADJI MOKHTAR - ANNABA

جامعة باجي مختار – عنابة

Faculté des sciences de l'ingénieur Année 2009
Département informatique

MEMOIRE

Présenté en vue de l'obtention du diplôme de MAGISTER

Intitulé :

Une Approche semi-automatique pour l'indexation de documents anciens

Option

Intelligence artificielle

Par

BENMOHAMED Abderrahim

DIRECTEUR DE MEMOIRE: Mokhtar SELLAMI Professeur U. BADJI Mokhtar

DEVANT LE JURY

PRESIDENT: Nadir FARAH Docteur U. BADJI Mokhtar

EXAMINATEURS: Messaoud RAMDANI Docteur U. BADJI Mokhtar
Hamid SERIDI Docteur U. GUELMA
Labiba MESLATI-SOUICI Docteur U. BADJI Mokhtar

Résumé

Actuellement, toutes les bibliothèques proposent une interface via le web permettant l'accès aux ressources numériques dont elles disposent. Ces dernières sont supposées numérisées, classées et catégorisées, indexées, et sauvegardées dans des formats bien définis. L'objectif global de notre travail de recherche est le développement d'une plate-forme permettant la sauvegarde, l'indexation et la recherche des documents anciens. Nous focalisons dans ce travail la présentation d'une nouvelle approche d'indexation et de recherche de documents anciens. La description retenue est une modélisation du document via une représentation du contenu en utilisant les descripteurs de contour.
Mots-clés : Anciens Manuscrits Arabes, Indexation de Documents, Recherche par le Contenu, chaîne de code, Traitement d’Images.

Abstract:
Currently, all the libraries propose a Web interface giving access to the numerical resources. These resources are supposed to be digitized, classified and categorized, indexed, and saved in well defined formats. The total objective of our research task is the development of a platform allowing the saving, the indexing and the research of the old documents. We focus in this work the presentation of a new approach of indexing and retrieval of old document. The description retained is a representation of the contents by using the descriptors of edge.
Key-words: Arabic Old Manuscript, Document indexing, Seek by Content, chain code, Image Processing.

ملخص

في الوقت الراهن ، تقدم جميع المكتبات عن طريق واجهة الويب برامج تتيح الوصول إلى الموارد الرقمية المتاحة لهم. هذا من المفترض مسحها ضوئيا وتصنيفها وفهرستها وتخزينها في صيغ محددة. الهدف العام من بحثنا هو وضع أرضية لحفظ، فهرسة واسترجاع الوثائق القديمة. علينا أن نركز في هذا العمل على تقديم نهج جديد للفهرسة والبحث عن الوثائق القديمة. الطريقة المنتهجة تتمثل في هيكلة محتوى الملفات عن طريق وصف لحدود الأشكال.

المخطوطات العربية السابقة، فهرسة الوثائق ، البحث عن طريق المحتوى ،سلسلة الرموز ، الكلمات الرئيسية :

Sommaire

Chapitre III indexation des documents anciens

Chapitre IV Détection et suivi des contours

Chapitre V Conception

Chapitre VI Implémentation et résultats

Introduction générale :

Les documents anciens représentent une richesse culturelle et scientifique inestimable. La sauvegarde de ce patrimoine qui souffre des outrages du temps et de la vicissitude des hommes, interpelle tant les scientifiques que les politiques pour préserver ces trésors des risques de détérioration voire de disparition.

Le potentiel véhiculé aujourd'hui par les technologies de l'information et de la communication, la maturité des systèmes de traitement d'images, la généralisation du web à haut débit, marquent un dans la conservation et l'exploitation de ces documents. Le défi consiste à préserver ces fonts par une numérisation massive et une série de traitements appropriés à l'aide d'algorithmes fiables et performants. L'hétérogénéité des pages, la dégradation de nombreux documents, sont des exemples reflétant la spécificité et les enjeux scientifiques que les chercheurs doivent surmonter.

Etroitement corrélé à cette thématique se pose le problème délicat de l'indexation des ces documents. Cette opération consiste à interpréter et reformuler un contenu documentaire pour le rendre accessible et exploitable par un système d'information. Cette opération d'indexation des documents anciens doit tenir compte de leurs forme,la mise en page, la variabilité de style et de fontes souvent complexes, l'absence de structure logique, la non homogénéité des caractères, mots et graphes souvent inconstants, l'inclinaison des textes etc.. Différentes méthodes sont exploitées donnant des résultats souvent mitigés, à titre d'exemple, l'approche structurelle construite à partir d'une étude des caractéristiques structurelles des documents anciens opère sans connaissance préalable du modèle de document et donne des taux de reconnaissance élevés, mais elle est lente et sensible aux inclinaisons.

L'approche qui semble la plus appropriée à notre avis est celle basée sur la comparaison des contours des mots. D'où notre proposition s'une approche semi automatique d'indexation et de recherche documentaire, où le choix des indexes se fait manuellement, tandis que leurs paramétrage (détection des primitives de bas niveau) et le stockage se fait automatiquement. Donc l'idée de base de notre approche est la recherche des indexes, détection de leurs contours, paramétrage des contours et stockage dans une base de données afin d'effectuer des recherches ultérieures. Le paramétrage des contours est effectué sous forme de chaîne de codes, tandis que la procédure d'optimisation permettra de résoudre partiellement le problème de taille des mêmes manuscrits ainsi que l'aspect des rotations.

L’algorithme de programmation dynamique DTW à permis d’obtenir des résultats très encourageants lors d’une première expérimentation menée au labo LRI, sur un échantillon de 1200 adresses postales labellisées envoyées au département informatique. Une deuxième expérimentation sur la base IFN/ENIT le taux de reconnaissance a été de 91.66% (sur un échantillon de 400 mots). Nous gardons à l’esprit cependant que le nombre de sous mots est un facteur qui n’aboutit pas à des résultats fiables, car il reste difficile de distinguer une différence de ponctuation et parfois même, la présence des coupures engendre des erreurs.

1. Introduction

Tout système de traitement d'image peut être vu comme une combinaison de deux phases, l'acquisition et le traitement proprement dit. La qualité des résultats dépend de l'algorithme mis en place et de son adéquation au problème posé, et aussi, de la qualité initiale des images traitées. Avant d'aborder le sujet de traitement d'image il faut savoir d'abord qu'est-ce qu'une image ? La définition du dictionnaire Larousse donne : «**image** : n.f. (lat. imago) Représentation imprimée d'un sujet quelconque ». En informatique, l'image est représentée sous format numérique**.** Une image «réelle » va être transformée en une image numérique par différents outils de transformation (caméra, scanner, satellite….). Ce chapitre présente un aperçu général du domaine du traitement des images. Il ne s'agit pas ici de détailler le domaine ni de présenter les récentes avancées technologiques mais plutôt une brève introduction.

2. Formation de l'image numérique

Une image est représentée par une matrice de dimension « nombre de ligne » x « nombre de colonnes ». Chaque élément de la matrice, nommé pixel, représente l'intensité lumineuse comprise entre 0 et 255, soit 256 niveau de gris, le niveau de gris 0 représente le noir tandis que le niveau de gris 255 représente le blanc. Autrement dit, une image est une forme discrète d'un phénomène continu obtenue après discrétisation. Cette forme est bidimensionnelle et les informations qui la présentent définissent les intensités lumineuses (couleurs ou niveaux de gris) [BER02].

I : [0, L-1] x [0, C-1] définit une image de L lignes et C colonnes dont l'information portée est définit dans un espace à p dimensions ;

Si I est une image en niveaux de gris, alors p = 1.

Si I est une image couleur, alors p = 3.

3. Amélioration de l'image

Supposons que l'on dispose d'une image numérisée qui contient des escaliers. Peut-on améliorer cette image par filtrage? Remarquons d'abord que toute image numérisée correspond à un signal: il se peut que l'effet d'escalier corresponde à la réalité du signal, et non pas à un effet de lissage. Toute technique de filtrage qui travaille à partir de l'image numérisée peut améliorer l'aspect visuel de l'image ou le détériorer, selon les caractéristiques originales de l'image [BER02]**.**

3.1 Traitement point à point

La technique de traitement la plus simple consiste à appliquer à chaque pixel de limage une fonction qui transforme son intensité. Cette fonction peut être représentée par une courbe de transfert (Figure I.1) qui donne en ordonnée l'image de chaque niveau d'intensité en abscisse :

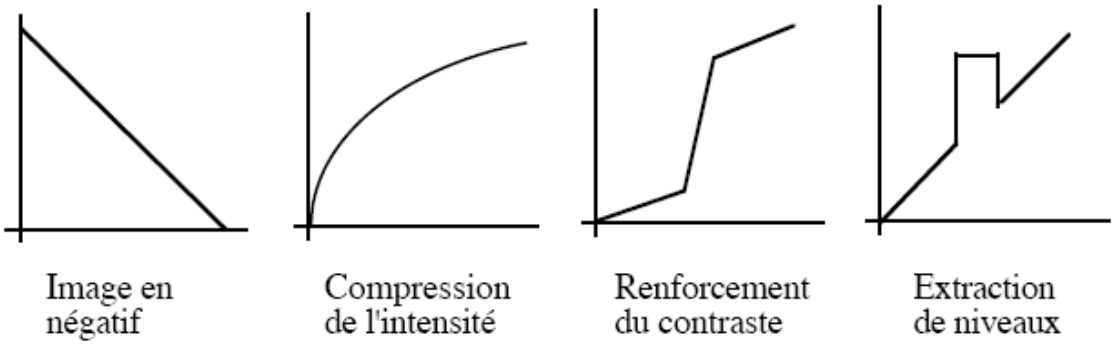


Figure I.1 - Exemples de courbes de transfert

La compression de l'intensité est utile pour des images qui ont une très large plage d'intensité, comme par exemple une transformée de Fourier. Le renforcement du contraste permet d'augmenter artificiellement le contraste d'une image terne.

L'extraction de niveaux permet de mettre en évidence les parties de l'image qui représente une intensité donnée.

Pour déterminer la courbe de transfert la mieux adaptée à un traitement donné il faut calculer l'histogramme de l'image qui représente l'ensemble des pixels en fonction de l'intensité. La technique d'égalisation de l'histogramme consiste à transformer l'image de telle sorte que l'histogramme soit plus plat, c'est-à-dire, il s'agit de rendre la fonction de densité de probabilité aussi uniforme que possible (Figure I.2). Pour obtenir ce résultat, il suffit de prendre comme fonction de transfert la fonction *f* suivante :

f (i) = (1/*n*) Somme i=0..i *H(i)* où:

n est le nombre total de pixels, *H(i)* le nombre de pixels d'intensité *i*, fourni par l'histogramme.

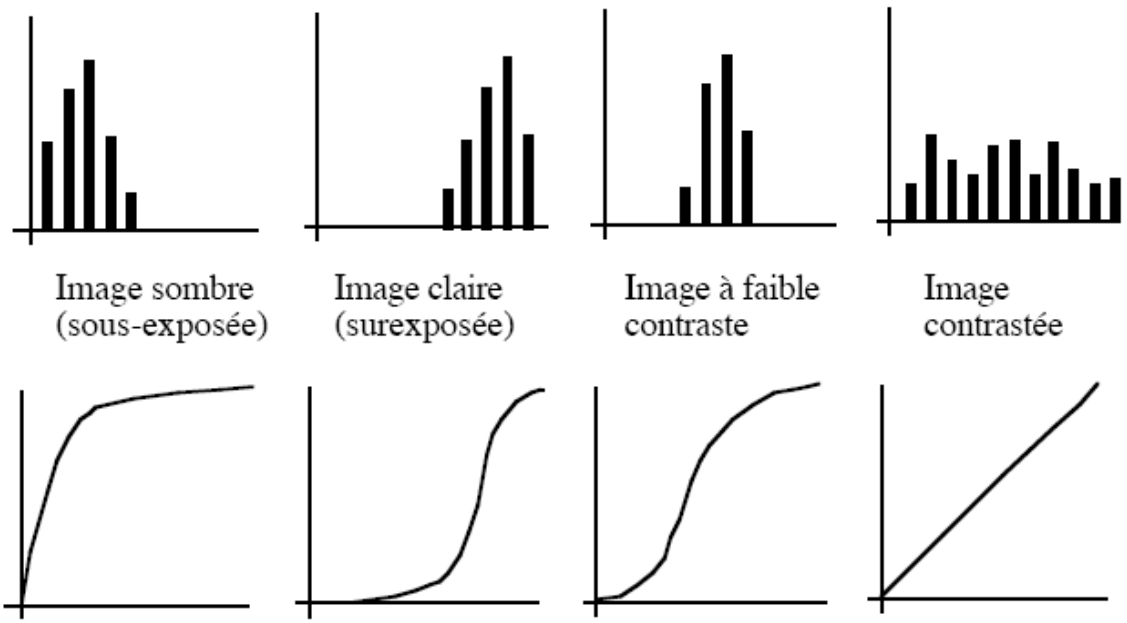


Figure I.2 - exemple d'histogrammes (nombre de points en fonction de l'intensité)

3.2 Méthodes locales

Les méthodes locales utilisent des techniques voisines des méthodes globales, mais en l'appliquant à un voisinage de chaque pixel. Ainsi, pour améliorer le contraste de l'image localement, on peut appliquer la technique d'égalisation d'histogramme à chaque pixel en utilisant les valeurs des pixels voisins.

Pour chaque pixel de l'image on calcule l'histogramme sur un voisinage (par exemple 7x7). On calcule aussi la fonction de transfert par égalisation de l'histogramme en appliquant cette fonction au pixel central.

Le calcul de l'histogramme peut être optimisé le long d'une ligne puisqu'à chaque déplacement, 7 pixels sortent du voisinage et 7 pixels entrent. De plus, il suffit de connaître la valeur de la fonction de transfert pour le pixel central et donc calculer la somme des 7x7/2 premiers histogrammes.

Une autre façon d'améliorer le contraste est de calculer la moyenne m et l'écart type s de la zone de l'image autour de chaque pixel et d'appliquer la transformation suivante:

$p' = A (p - m) + m$

$A = k M / s$

Où:

p est l'intensité du pixel, p' l'intensité après transformation, M la moyenne globale des intensités et k un réel entre 0 et 1

A étant inversement proportionnel à la variance autour du pixel considéré, il est plus grand dans les zones de faible contraste. Le terme A (p-m) renforce donc le contraste dans ces zones. En pratique, il est parfois nécessaire de borner A entre deux valeurs Amin et Amax.

4 Binarisation des images

La binarisation ou le seuillage est la technique de classification la plus simple où les pixels de l'image sont partagés par un seul seuil s en deux classes : ceux qui appartiennent au fond et ceux qui appartiennent à l'objet. L'image est alors séparée en deux classes de façon à ce que l'information comprise entre 0 et s est retenue et l'autre non, ou vice-versa.

Soit l'image I (M x N), supposons que f (x, y) représente le niveau de gris du pixel aux coordonnées (x, y), s est le seuil choisi, les pixels de l'objet sont ceux ayant le niveau de gris inférieur à s et les autres ayant le niveau de gris supérieur à s sont des pixels du fond. Alors, l'image binarisée G est déterminée par les pixels (x, y) dont la valeur est:

$$g(x,y)= \begin{cases} 1 \text{ si } f(x,y) > s \\ 0 \text{ si } f(x,y) <= s \end{cases} \qquad (1)$$

Il existe trois grandes techniques de sélection du seuil s :

Global, local et adaptatif [OTS79]. Comme il y a des différentes façons de déterminer le seuil s, il peut être considéré comme une fonction sous forme de s = t((x, y), p(x, y), f (x, y)) où p(x, y) représente des propriétés locales du point (x, y). Si s ne dépend que de la valeur f*(x, y)* du point, le seuil est *global*, s'il dépend en plus de p(x, y), s est un seuil *local*. Et si s dépend à la fois de (x, y), de p(x, y) et de f(x, y), on dit un seuil adaptatif ou dynamique.

Parmi les méthodes de binarisation les plus utilisées dans le traitement des documents historiques nous avons:

4.1 Binarisation efficace des documents historiques :

Cette méthode combine plusieurs méthodes de binarisation d'où $R_1 (x,y)$, $R_2 (x, y)$,…$R_n (x,y)$ représentent leurs résultats après une application sur l'image IF(x,y), n est un nombre impair, n=2m+1. [GAT08]

Chaque image binarisée R_i est définie par :

$$R_i (x,y)= \begin{cases} 1 \text{ le trait} \\ 0 \text{ le fond} \end{cases} \qquad (2)$$

L'image binaire finale Br est calculée comme suit :

$$B_r(x,y) = \begin{cases} 1 \text{ si } \sum_{i=1}^{2m+1} R_i(x,y) > m \\ 0 \text{ sinon} \end{cases} \quad (3)$$

4.2 Méthode à seuillage adaptatif

Cette méthode est à seuillage adaptatif, et appliquée à des documents, au niveau de gris [CHE08], et de très faible qualité. Le seuil local est calculé en fonction de la moyenne locale « *moy* », l'écart type et le niveau de gris minimum local.

Le seuil est calculé comme suit :

1- Calculer la moyenne "*moy"* pour chaque pixel : $moy = \frac{\sum_{i=1}^{N} Ii}{N}$ (4)

Où N est le nombre de pixel dans un voisinage local et Ii la valeur de l'intensité du pixel i.

2- Calculer l'écart type s : $s = \sqrt{(1/N) \times \sum_{i=1}^{N} (I_i - moy)^2}$ (5)

3- Calculer le niveau de gris minimum M

4- Calculer Rs est le rang dynamique de l'écart type du niveau de gris local.

5- le seuil T est donné par :

$T = (1-a_1) \times moy + k_1 \times (s/Rs)^3 \times (moy-M) + k2 \times (s/Rs)^2 \times M$ (6)

Où: $0.15 \leq a_1 \leq 0.25$, $0.05 \leq k_1 \leq 0.15$ et $0.01 \leq k_2 \leq 0.05$.

5 Morphologie mathématique

5.1 Erosion : Cette opération est très importante avec la dilatation. Son principe est le suivant : soit un ensemble B, repéré par son centre p, B appartient à l'ensemble X et représente une forme quelconque. B est déplacé de telle sorte que son centre p occupe toutes les positions de l'espace. Est ce que l'ensemble B est entièrement inclus dans X pour chaque

position de p ? Donc l'ensemble des p répondant à cette question forme une érosion de X par B notée : X Θ B [DRO94]

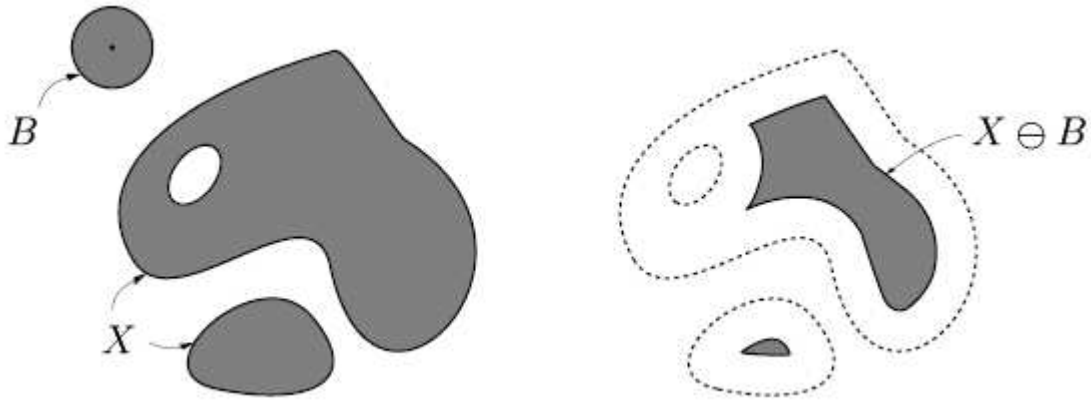


Figure I.3 Erosion de X par B

5.2 Dilatation : L'opération de dilatation se définit d'une manière analogue, on obtient le dilaté de X par B en prenant la totalité de la surface de couverture lorsque l'origine de B occupe successivement toutes les positions dans X. Elle est notée par X $\oplus$ B [BEU90].

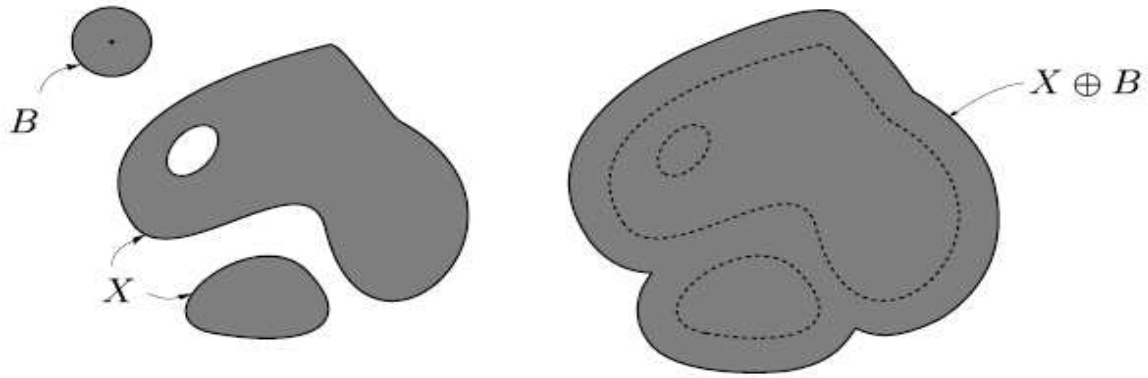


Figure I.4 Dilatation de X par B

5.3 L'ouverture: Les opérations "Erosion" et "Dilatations" peuvent être combinées et une opération d'érosion suivie d'une dilatation s'appelle une ouverture et notée par X ° B

Où : X ° B = : (X Θ B) $\oplus$ B

5.4 La fermeture : L'opération de fermeture est obtenue en inversant l'ordre des opérations utilisées pour définir l'ouverture, elle notée par X ● B, Où :

X ● B = : (X $\oplus$ B) Θ B

6 Segmentation

La segmentation des images consiste à regrouper les pixels qui ont les mêmes propriétés afin de former des régions connexes [GUI03]. Il existe deux familles d'approches que l'on peut faire coopérer : les approches contours et les approches régions. Dans les approches contours les régions sont déterminées par leurs contours par contre dans les approches régions d'autres critères font intervenir tels que l'agrégation de pixels et l'homogénéité.

6.1 Segmentation contour

L'extraction du contour consiste à rechercher les points de contour, réduire le contour à un pixel, fermer les contours ouverts, et enfin, coder le contour [DER87].

* **L'approche gradient :** consiste à calculer la dérivée première en point de coordonnées p,q comme suit :

$$\nabla f = \begin{bmatrix} Gx \\ Gy \end{bmatrix} = \begin{bmatrix} \partial f / \partial x \\ \partial f / \partial y \end{bmatrix} \qquad (7)$$

$$\text{Module} = |\nabla f| = \sqrt{Gx^2 + Gy^2} \qquad (8)$$

$$\text{Direction} = \arg(\nabla f) = \tan^{-1}\left[\frac{Gy}{Gx}\right] \qquad (9)$$

Ces formules peuvent être transformées en opérations matricielles en utilisant l'un des quatre masques de convolution suivants : [CHR06]

-1	-1	-1
0	0	0
1	1	1

Dérivées selon y

-1	0	1
-1	0	1
-1	0	1

dérivée selon x

-1	-2	-1
0	0	0
1	2	1

filtre de Prewit

-1	0	1
-2	0	2
-1	0	1

filtre de Sobel

Il est possible d'avoir des tailles de masques plus grandes qui permettent d'améliorer la qualité d'estimation de la dérivée mais par conséquence augmentent le temps de calcul [CAN86].

Notons que la direction du gradient en un point est perpendiculaire à la direction du contour en ce point. À partir des valeurs du module, une extraction des valeurs importantes permet de

ne retenir que les variations de niveaux de gris ou de couleur significatives (supérieures à un seuil donné), donc à un contour. Une variante pour ce calcul du gradient en x et en y est le gradient directionnel, appelé aussi gradient boussole, qui consiste à convoler l'image avec huit masques obtenus par rotations successives de $\pi/4$ du masque de base, indiquant ainsi les huit directions possibles pour un contour.

* **L'approche Laplacien:** Ces méthodes ont été proposées en 1976. Elles ont eu une grande importance historique, étant considérées comme le prototype du détecteur de contour inspiré des systèmes biologiques (*primalsketch* de Marr) [MAI03]. Elles utilisent le fait que le passage par zéro du Laplacien permet de bien mettre en évidence les extremums de la dérivée :

$$\left.\frac{\partial^2 f}{\partial^2 x}\right|_{p,q} \approx \frac{1}{4\lambda x^2}\,[f(p+1,q) - 2f(p,q) + f(p-1,q)] \qquad (10)$$

$$\left.\frac{\partial^2 f}{\partial^2 y}\right|_{p,q} \approx \frac{1}{4\lambda y^2}\,[f(p,q+1) - 2f(p,q) + f(p,q-1)] \qquad (11)$$

Ces formules peuvent être transformées en opérations matricielles en utilisant l'un des deux masques de convolution suivants :

0	-1	0
-1	4	-1
0	-1	0

-1	-1	-1
-1	8	-1
-1	-1	-1

L'extraction des contours grâce aux dérivées secondes se fait à partir d'une seule convolution et n'exploite pas les maxima mais les passages par zéro de la fonction bidimensionnelle. Cependant, l'opérateur Laplacien est sensible au bruit et ne donne aucune indication quant à la direction des contours.

* **Filtre de Canny :** cette approche à donnée une bonne détection de contour et a conduit à

des détecteurs de très bonne qualité, la méthode de Canny assure également une très bonne localisation des contours [MAI03]**,** et que chaque contour n'aura qu'une seule détection. Le filtre a pour réponse impulsionnelle $\psi(x)$, pour avoir une bonne détection on a

$$\square = \frac{\int_{0}^{\infty} \psi(x)\, dx}{\int_{-\infty}^{\infty} \psi^2(x)\, dx} \qquad (12)$$

Pour avoir une bonne localisation :

$$A = \frac{|\psi^2(0)|}{\sqrt{\int_{-\infty}^{\infty} \psi^2(x)\, dx}} \qquad (13)$$

*** Filtre de Deriche :** Le détecteur de Deriche répond exactement aux mêmes critères de qualité que celui de Canny, mais possède une réponse impulsionnelle infinie. Il a pu donc être synthétisé de façon récursive particulièrement efficace Le filtre de Deriche a une expression générale de la forme :

$$\psi(x) = -cx \exp(-\alpha|x|) \qquad (14)$$

Avec **:**

$$C = \frac{|1- \exp(-\alpha)|^2}{\exp(-\alpha)} \qquad (15)$$

6.2 Segmentation région : Fondamentalement, la segmentation est un processus qui consiste à découper une image en régions connexes présentant une homogénéité selon un certain critère, comme par exemple la couleur. L'union de ces régions doit redonner l'image initiale. La segmentation région est une étape importante pour l'extraction des informations qualitatives de l'image. Elle fournit une description de haut niveau : chaque région est connectée à ses voisines dans un graphe et chaque région porte une étiquette donnant des informations qualitatives; taille, couleur, forme, orientation. L'image se réduit donc à un graphe de nœuds étiquetés qui contient presque toutes les informations utiles au système. Les

arcs de ce graphe précisent si les deux régions connectées sont en simple contact ou si l'une est incluse dans l'autre.

D'autres informations topologiques ou spatiales peuvent également être stockées comme par exemple le fait qu'une région est au dessus d'une autre. Selon les techniques de segmentation utilisées, la construction de ce graphe peut être plus ou moins complexe. Parmi les techniques de segmentation en régions nous avons :

- **Croissance de région par agrégation de pixels :** Les méthodes d'accroissement de région sont les méthodes de segmentation les plus simples. L'algorithme part de petits éléments de l'image qu'il va tenter de regrouper en éléments plus importants. Supposons une région de couleur homogène R. Initialement, R = {un pixel}.

On va étendre la région R en incluant les pixels situés sur la frontière et dont la couleur est proche de celle de R[1].

En répétant cette procédure jusqu'à ce qu'il n'y ait plus de pixels de couleur assez proche sur la frontière, on obtient une région de couleur homogène maximale autour du pixel de départ. La région initiale accroît en absorbant des pixels de la frontière, jusqu'à stabilité par rapport à cette propriété d'homogénéité. Cette méthode présente deux limitations sévères qui n'en font pas une méthode très efficace :

1. Les régions obtenues dépendent fortement des pixels d'amorçage choisis et de l'ordre dans lequel les pixels de la frontière sont examinés.
2. Le résultat final est très sensible à la valeur du seuil δ.

Pour illustrer les problèmes, considérons trois pixels adjacents a, b et c dont les intensités respectives sont 8,10 et 11 (par exemple, le niveaux de gris). Le seuil est 2. La région initiale est constituée du pixel b. Deux schémas de regroupement pour les points frontières a et c sont possibles **(voir Figure I.5).**

Le pixel central b est l'amorce. Compte tenu du seuil δ = 2, a et c sur la frontière devraient être ajoutés à l'amorce b. Si l'on commence par tenter d'agréger a, le résultat du regroupement, noté [ab], a pour intensité moyenne 9 et c s'y ajoute ensuite puisque δ< |9-11|. On a donc regroupé a, b et c. Si maintenant l'algorithme commence par examiner le point frontière c, le groupement de b et c donne [bc] dont l'intensité est 10,5. Le point frontière 'a' d'intensité 8 est trop éloigné et il est considéré comme appartenant à une autre région. On obtient donc deux groupements au lieu d'un.

[1] La variation de couleur est inférieure à un certain seuil

Ce petit exemple illustre combien l'ordre d'examen des points sur la frontière peut influencer sur le résultat de l'algorithme. Par ailleurs, comme nous allons le voir, le résultat final dépend très sensiblement du seuil. Une petite variation de ce seuil peu conduire à des modifications importantes.

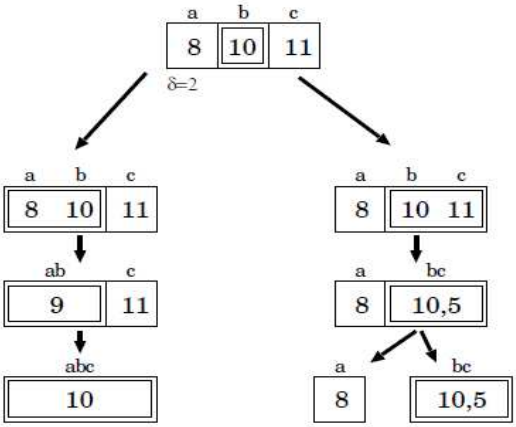


Figure I.5. Deux schémas de regroupement pour des pixels a, b et c

*** Méthodes du type division/fusion "split and merge" :**

L'algorithme « split and merge » a été présenté la première fois en 1974 par Pavlidis et Horowitz [HOR77]. Cet algorithme s'apparente dans son principe à l'algorithme d'accroissement de région que nous venons de présenter. La différence principale provient de la nature des régions élémentaires agrégées. Dans l'algorithme « split and merge », les régions agrégées proviennent d'une première phase (split) de traitement de l'image qui construit de manière récursive des régions carrées de taille variable mais homogènes.

La méthode de découpage de l'image utilisée dans cet algorithme est basée sur la notion de *quadtree* Cette structure de données est un arbre quaternaire qui permet de stocker l'image à plusieurs niveaux de résolution. On part d'une région initiale qui est l'image entière. Si cette image arrive à avoir un critère d'homogénéité de couleur, l'algorithme s'arrête. Sinon, on découpe cette région en quatre parties de même taille et on lance la procédure récursivement dans ces quatre parties. La région initiale va être stockée comme un noeud dans un graphe et les sous parties comme des fils de ce nœud. La figure I.6 montre une image en noir et blanc 8x8 et le découpage correspondant à chaque niveau.

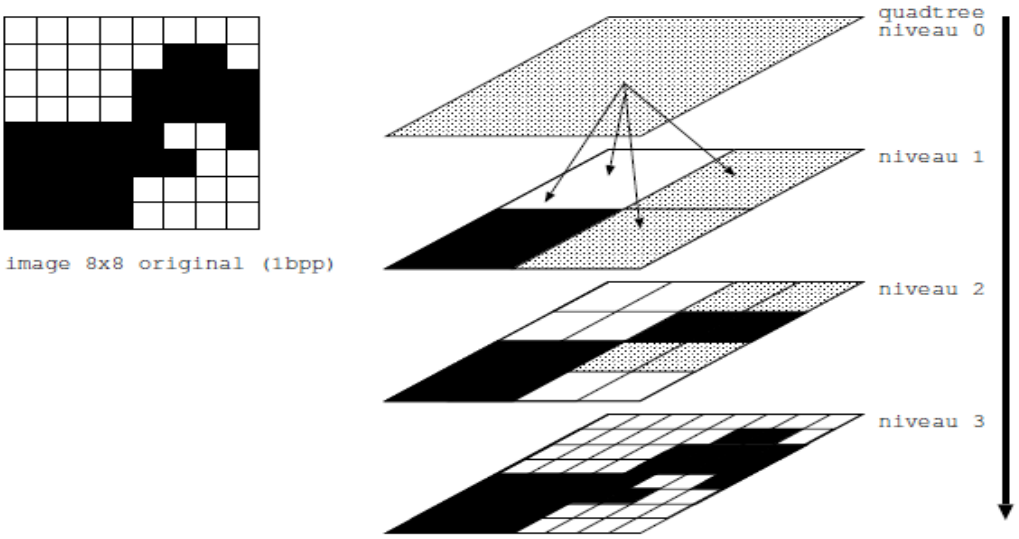


Figure I.6 découpage par « quadtree » d'une image 8 x 8

Dans cet exemple, le critère d'homogénéité est absolu: une zone est dite homogène si elle ne contient que des pixels de même couleur (seuil=100%). On peut être plus tolérant et accepter qu'une zone soit déclarée homogène dès que plus de 75% d'une couleur domine.

Le critère d'homogénéité est fixé par un seuil sur la variance de la couleur dans la zone en cours d'examen. Au dessus de ce seuil, la zone est découpée en quatre, en dessous, elle est conservée et constitue un nœud terminal de l'arbre. On lui attribue alors la couleur de la moyenne des pixels la constituant.

Pour réaliser cette fusion, il faut d'abord tenir à jour une liste des contacts entre régions (chaque région dispose d'un pointeur sur une liste chaînée de régions avec lesquelles elle est en contact). On obtient ainsi un graphe d'adjacence de régions. Ce graphe de contact doit se construire en même temps que l'arbre de découpage. Ensuite, l'algorithme va marquer toutes les régions comme « non traitées » et choisir la première région R non traitée disponible.

Les régions en contact avec R sont empilées et sont examinées les unes après les autres pour savoir si elles doivent se fusionner avec R. Si c'est le cas, la couleur moyenne de R est mise à jour et les régions en contact avec la région fusionnée son ajoutés à la pile des régions à comparer avec R. La région fusionnée est marquée « traitée » Une fois la pile vide, l'algorithme choisi la prochaine région marquée « non traitée » et recommence, jusqu'à ce que toutes les régions soient traitées

6.3 Coopération entre détection de contour et détection de région :

L'idée principale des approches coopératives est de lier une approche contour avec une approche région au lieu d'utiliser l'une ou l'autre séparément. Une des approches coopératives est celle de [WRO87]**,** son principe est de contrôler un processus hiérarchique de fusion de régions homogènes avec une carte de contour préalablement calculée, le fonctionnement de la méthode se fait en quatre étapes :

1 – détermination d'une carte de contour : la détermination des contours se fait par une méthode de morphologie mathématique, différence de filtres récursifs, filtre de Canny…etc.

2 – détermination d'une segmentation initiale

3- création d'un graphe d'adjacence des régions : ce graphe devra contenir toutes les informations nécessaires pour la suite de traitement

4 - regroupement des régions

6.4 Segmentation multi-échelle:

Nous allons présenter ici un résumé du principe de multi-résolution et de la décomposition en ondelettes. La multi-résolution est obtenue en utilisant une transformation en ondelettes [LAN05]. La transformation en ondelette est obtenue par projection orthogonale de l'image sur des espaces affines obtenus à partir de la base d'ondelette. On parle d'espace d'approximation (V) et d'espace de détail (W). L'espace d'approximation de niveau i contient des signaux plus grossiers que l'espace d'approximation de niveau i+1.

Plus i augmente, plus on s'intéresse aux basses fréquences du signal. L'espace de détail de niveau i est la différence d'information entre l'espace d'approximation de niveau i-1 et l'espace d'approximation de niveau i (il sert donc à stocker des hautes fréquences).

Les valeurs des images d'approximation et de détail sont calculées en utilisant un paramètre de dilatation/réduction σ et un paramètre de translation τ. Physiquement, σ permet d'analyser le signal à différentes fréquences et τ de parcourir le signal $W\{f, \psi\}(\sigma, \tau)$

W est la valeur du coefficient d'ondelette obtenue par application de l'ondelette ψ au signal *f* avec comme coefficient de dilatation σ et comme coefficient de translation τ.

$$W\{f, \psi\}(\sigma, \tau) = (f(t), \psi_{\sigma, \tau}(t)) \qquad (16)$$

Avec :

$$\psi_{\sigma, \tau}(t) = \frac{1}{\sqrt{|\sigma|}} \psi\left(\frac{t - \tau}{\sigma}\right) \qquad (17)$$

Où $\psi(t)$ est appelée l'ondelette mère.

L'ondelette utilisée ici est bi-orthogonale. Les filtres utilisés pour la décomposition et la reconstruction du signal à partir de la transformation en ondelettes ne sont donc pas identiques.

Parmi les ondelettes bi-orthogonales, on trouve les fonctions splines. Par ailleurs, une transformation en ondelettes discrète dyadique signifie que le coefficient de dilation utilisé est 2. Le coefficient de dilation correspondant au niveau i est donc $\sigma i=2\text{-}i$.

Ce sont donc les images d'approximation obtenues à partir de ces coefficients de dilatation qui vont composer la multi-résolution.

L'image I (équivalente à l'approximation de niveau 0 V0) est donc décomposée en n niveaux d'approximation ($\{V_i\}$, $1\leq i\leq n$) et de détails ($\{W_{xi}, W_{yi}\}$, $1\leq i\leq n$) :

$$I = V_1 + W_{x1} + W_{y1} \qquad (18)$$

$$Vi = W_{xi+1} + W_{yi+1} + V_{i+1},\ 0\leq i\leq n \qquad (19)$$

La valeur de n dépend de la taille de l'image. On considère par la suite des images I de 2n lignes et 2n colonnes.

$$I = \{ (x_{ij},y_{ij}) \mid i \in [0,2n-1],\ j \in [0,2n-1] \} \qquad (20)$$

6.5 Segmentation par ligne de partage des eaux :

Enfin, la segmentation par Ligne de Partage des Eaux (LPE ou watershed en anglais). Elle est basée sur l'analogie entre une image en niveaux de gris et une surface topographique. Ce relief comporte un certain nombre de structures topographiques (dômes, vallées, lignes de crêtes) qui sont utiles pour nous renseigner sur le contenu de l'image. Plus particulièrement, les lignes de crêtes ou lignes de partage des eaux sont assimilées aux contours de l'image. Retrouver les structures contenues dans l'image revient alors à rechercher les bassins versants c'est-à-dire, en reprenant l'analogie hydrologique. Une zone géographique pour laquelle toute goutte d'eau, suivant la ligne de plus grande pente (ligne de crête) arrivera dans un même minimum. Tous les bassins de l'image sont connexes et leur ensemble forme une partition de l'image. La frontière de ces bassins constitue la ligne de partage des eaux ou LPE. Une méthodologie courante pour la détermination des LPE d'une image est basée sur un processus d'immersion de l'image. Dans les applications de segmentation d'images, cette approche utilise généralement la norme du gradient de l'image initiale comme point d'entrée. En effet, l'objectif étant de déterminer des régions homogènes, ce sont des zones de fort gradient qui doivent constituer les frontières (les lignes de crêtes recherchées). Cette image de gradient est inondée de manière récursive par ses minima locaux. Lorsque deux bassins versants se

rejoignent et risquent de se mélanger, on construit une digue. Lorsque le relief est totalement immergé, les digues ainsi formées constituent les lignes de partage des eaux.

Les algorithmes de calcul de la ligne de partage des eaux les plus efficaces ont une complexité qui dépend linéairement du nombre de pixels de l'image, ce qui les classe parmi les méthodes de segmentation les plus rapides. Ceci les rend particulièrement attrayants dans des applications traitant des images de grande taille. Par ailleurs, les contours obtenus sont systématiquement fermés et ont un positionnement précis en correspondance avec les objets à détourer. Cependant, cette approche a tendance à générer une sur-segmentation importante de l'image. De nombreuses solutions à ce problème ont été proposées comme, par exemple, l'introduction de marqueurs (on ne s'intéresse alors qu'à certains minima locaux lors du processus d'immersion), l'ajout de prétraitement de type filtrage passe bas, l'ajout d'étape de fusion de régions.

7. Texture

La texture est une caractéristique importante pour la segmentation de plusieurs types d'images [KUN93], des images aériennes aux images médicales. Le système visuel humain est extrêmement performant dans ce domaine. Ainsi, sur une scène naturelle en extérieur on distingue sans difficulté les différentes textures : l'herbe, le feuillage, le sable,…etc.

Cependant, la segmentation automatique d'images par analyse de texture est toujours l'un des problèmes les plus difficiles dans le domaine de traitement d'images; problème qui reste d'ailleurs ouvert car il n'y a pas de définition universellement acceptée de la notion de texture. Intuitivement, la notion de texture est liée à l'aspect homogène d'une surface. Une propriété essentielle de la perception texturelle est son invariance par translation. Une texture laisse la même impression au système visuel, quelle que soit la partie de la texture qui est observée.

Nous nous situons ici dans le cadre de la classification supervisée de textures, c'est-à-dire dans le cas où l'on a prédéfini les classes (par exemple : ciel, terre, habitations, ...etc.), et où l'on dispose a priori d'échantillons représentatifs de ces textures. Il existe également des méthodes de classification non supervisée, qui segmentent une image en zones, le regroupement des pixels de l'image dans une même zone se faisant sur un critère de proximité dans l'espace paramétrique. Le résultat obtenu (segmentation de l'image) dépend alors uniquement du choix des paramètres. Les méthodes non supervisées peuvent être utiles pour découper une image en zones, et servir de prétraitement à des étages supérieurs (reconnaissance d'objets, ...etc.). Mais les zones ne correspondent pas obligatoirement à des entités physiques (ciel, feuillage, ...etc.). Des méthodes non supervisées performantes exploitant des modèles des frontières ou des régions ont été publiées récemment.

Un grand nombre de méthodes a été proposé pour la discrimination de textures. Toutefois, il est possible de les regrouper en deux grandes familles :

1. Les méthodes statistiques
2. Les méthodes structurelles

Les applications sont très variées, et couvrent divers types d'images, des images médicales aux images aériennes. Le point fondamental dans toutes ces méthodes est le choix d'un ensemble de caractéristiques qui permettent de réduire la dimension des données à une quantité acceptable pour le classifieur, tout en tentant de préserver une grande partie de l'information discriminante. De plus, pour la plupart des classifieurs, il est souhaitable que la loi de probabilité de ces caractéristiques conditionnellement à la classe soit simple (par exemple de type gaussien). Les méthodes statistiques considèrent la texture comme un champ aléatoire à deux dimensions, et les caractéristiques requises sont obtenues par des approches telles que les matrices de cooccurrence, la fréquence spatiale, la corrélation, ou les modèles paramétriques. Quelques études comparatives non exhaustives ont été proposées. Elles montrent que les méthodes basées sur les matrices de co-occurrence sont généralement les plus performantes, bien que des méthodes moins gourmandes en puissance de calcul puissent donner des résultats comparables sur des ensembles réduits de textures.

Les méthodes structurelles caractérisent la texture par des primitives élémentaires appelés « texels », et par l'arrangement spatial de ces primitives. Bien que saisissant certains aspects du fonctionnement du système visuel humain, les méthodes structurelles sont généralement plus complexes que les méthodes statistiques, et réagissent assez mal en présence de textures faiblement structurées telles que l'herbe, le feuillage, la laine, et un grand nombre de textures naturelles.

On constate donc qu'aux deux grandes familles précédemment citées correspondent deux définitions de la texture une définition statistique et une définition structurelle.

Une définition structurelle semble mal adaptée au cas des images Infrarouge, qui sont souvent floues, bruitées, avec des contours mal définis. Ceci est moins vrai pour les images naturelles dans le domaine visible. Toutefois, un grand nombre de textures naturelles peuvent être vues comme des champs aléatoires.

Signalons enfin des travaux intéressants visant à segmenter des images selon la texture en utilisant des réseaux neuro-mimétiques qui imitent la structure du système visuel humain. Ces travaux à long terme sont pour l'instant validés sur des configurations simples. D'autres travaux dans le domaine des réseaux de neurones consistent à alimenter un classifieur neuronal avec des paramètres classiques.

8. Conclusion: Dans ce chapitre nous avons présenté quelques définitions des méthodes utilisées dans le traitement de l'image, qui est un domine très vaste. Pour cela nous avons expliqué les principes des méthodes qui doivent être appliquées sur les images avant de procéder à l'indexation, notamment : l'amélioration de l'image, la binarisation et la segmentation. Cette panoplie de méthodes nous offre la possibilité d'utiliser celles qui sont plus efficaces et facilement implémentables.

1. Introduction

Le nombre et la taille des entrepôts d'images ne cessent de croîtrent à une vitesse extraordinaire avec la disponibilité grandissante et le prix décroissant des appareils de captures (caméra, scanner, appareil photo, téléphone mobile, PDA, …). Ces bases de données doivent être structurées, organisées, étiquetées, et indexées afin de rendre leurs contenus accessibles à un large public via des moteurs de recherche. L'indexation est une étape indispensable et importante dans tout système de traitement et de diffusion d'images.

Plusieurs approches et techniques d'indexation ont été développées au cours des dix dernières années. Si l'approche manuelle, qui consiste à associer manuellement aux images des descripteurs plus ou moins sémantiques, est la plus fiable, les chercheurs préfèrent plutôt les approches automatiques et semi-automatiques.

Dans ce chapitre nous allons présenter un aperçu des principales techniques utilisées pour la description et l'indexation des images. Nous commencerons par exposer les caractéristiques extraites des images et utilisées pour l'indexation, ensuite nous abordons les techniques de classification et nous terminons par les techniques d'indexation proprement dites.

2. Caractéristiques des images

2.1. La couleur

C'est le premier descripteur qui est employé pour la recherche d'images [SME00]. Une technique très utilisée pour la couleur est l'intersection d'histogrammes [BOU05]**.** Les histogrammes sont faciles à calculer, et robustes à la rotation et à la translation. Cependant l'utilisation d'histogrammes pour l'indexation et la recherche d'images pose quatre problèmes. Premièrement, ils sont de grandes tailles, donc par conséquent il est difficile de créer une indexation rapide et efficace en les utilisant tels quels. Deuxièmement, ils ne possèdent pas d'informations spatiales sur les positions des couleurs. Troisièmement, ils sont sensibles à de petits changements de luminosité, ce qui est problématique pour comparer des images similaires, mais acquises dans des conditions différentes. Et quatrièmement, ils sont inutilisables pour la comparaison partielle des images, puisque calculés globalement sur toute l'image. Pour remédier à ces problèmes, deux approches sont possibles.

La première approche ajoute des informations spatiales aux histogrammes. Tels que les moments d'inertie et la cohérence spatiale. Un pixel est cohérent s'il appartient à une région validée par la segmentation et incohérent autrement, la comparaison entre deux histogrammes devient la comparaison entre les valeurs d'histogrammes dans les classes correspondantes. La deuxième approche recherche d'autres espaces de couleurs qui se basent sur la perception de couleur de l'humain. L'espace RVB est un espace de couleur utilisé

couramment, dans tous les systèmes de vision automatique, mais il n'est pas forcément le mieux adapté. En effet, les trois composantes RVB sont très dépendantes les unes des autres. Un simple changement d'éclairage dans la scène modifie les trois composantes, alors que les objets de la scène n'ont pas changés de couleur, mais se sont simplement assombris ou éclairés.

2.2 La texture : La texture, autre primitive visuelle, est étudiée depuis une vingtaine d'années et plusieurs techniques ont été développées pour l'analyser. Une méthode très connue pour analyser la texture est la matrice de co-occurrences. Afin d'estimer la similarité entre des matrices de co-occurrences, quatre caractéristiques extraites de ces matrices sont largement utilisées: l'énergie, l'entropie, le contraste et le moment inverse de différence. Il existe aussi d'autres méthodes pour analyser les textures dont celle basée sur les filtres de Gabor. Après avoir appliquée la transformation de Gabor sur une image, une région de texture est caractérisée par la moyenne et la variance des coefficients de transformation.

2.3 Les contours (transformée en ondelettes) :

La transformée en ondelettes permet de capturer la régularité présente le long des contours rectilignes. Pour ce faire, on représente une image de façon bijective dans le domaine polaire. On obtient ainsi une conversion des singularités rectilignes dans le domaine spatial en singularités ponctuelles dans le domaine de Radon. Cela peut se traduire pour une image $f \in L^2(R^2)$ en : $Rf(\theta,t) = \int_{R^2} f(x,y)\,(x\cos(\theta)+y\sin(\theta)-t)\,dxdy$ (22) Où :

$R_f(\theta, t)$ représente la projection radiale de f sur la droite d'équation $x\cos(\theta) + y\sin(\theta) = t$.

La transformée en ondelettes s'obtient alors en appliquant une transformée en ondelettes 1D le long de $R_f(\theta, .)$ en utilisant la variable d'intégration t. La décomposition de f s'écrit alors :

$$RT_f(a,b,\theta) = \frac{1}{\sqrt{a}} \int_R \psi\left(\frac{t-b}{a}\right) R_f(\theta,t)\,dt \qquad (23)$$

Où 'a' est un facteur d'échelle, 'b' un paramètre de translation, 'θ' l'angle de projection et 'ψ' une ondelette.

2.4 Calcul de l'orientation et de la courbure multi échelle : La transformée en ondelettes permet d'obtenir pour chaque pixel des contours d'un objet de l'image un nombre de coefficients caractéristiques dépendant du niveau d'échelle et du nombre d'angles associés. Pour chacun de ces coefficients on évalue leur représentativité en comparaison avec les autres coefficients correspondants au même niveau d'échelle et à la même analyse angulaire. Pour un pixel p de l'image *f*, si on constate que le coefficient qui lui correspond c(n, o), pour une

échelle n et un angle o, est significatif, alors on sauvegarde l'orientation o dans une liste l d'orientations associée au pixel p. En reproduisant cette analyse sur toutes les orientations pour tous les pixels, on obtient une caractérisation angulaire de chaque pixel du contour d'un objet. Voir figure II.1

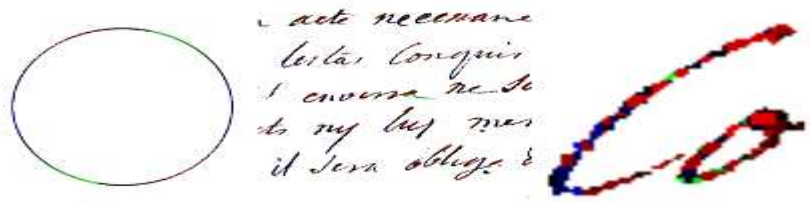


Figure II.1 Evaluation des orientations sur un extrait de manuscrit

D'après [JOU07] pour un pixel p d'une courbe L, plus le nombre d'orientations considérées comme significatifs pour le pixel p est important, plus la courbure de L est élevée en ce point; le cas extrême étant atteint dans le cas d'un pixel isolé. Voir figure II.2

Figure II.2 Evaluation des courbures sur un extrait de manuscrit

2.5 Calcul de la signature :

Les deux primitives retenues pour le calcul de la signature des extraits manuscrits sont la courbure et l'orientation sur les contours des formes d'une image. Le calcul de la signature est le suivant. Il s'agit d'une signature 2D avec en abscisse une échelle de la courbure et en ordonnée une échelle des orientations. Pour chaque pixel p on parcourt sa liste d'orientations *l*. Pour chaque élément o de *l* on forme un couple (c, o), c étant la courbure correspondante à p et donc la longueur de *l*. On incrémente ensuite la valeur correspondante à ce couple dans la signature.

Pour calculer la distance entre deux signatures nous pouvons utiliser la fonction S(X,Y) définie comme suit :

Cov(X,Y)

(24)

$$S(X,Y) = \frac{}{\sigma x \, \sigma y}$$

où Cov(X, Y) est la covariance entre X et Y et σx et σy sont les écarts-types.

.

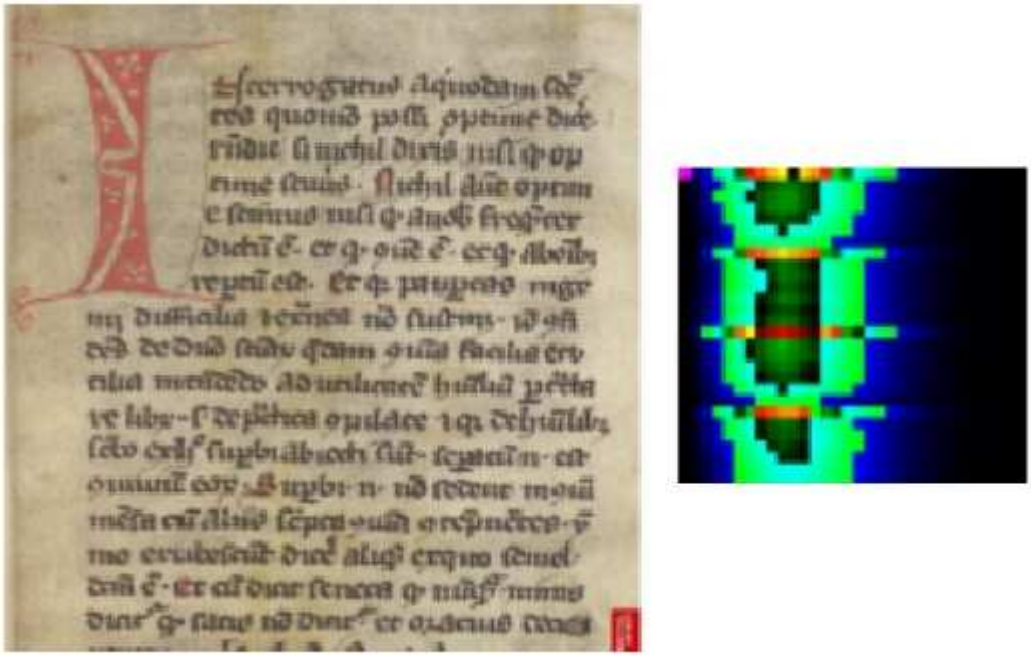


Figure II.3 Signature de manuscrit

2.6 Le gradient couleur : ce descripteur est utilisé pour représenter les informations globales de l'image, il est obtenu par le calcul du maximum de l'histogramme normalisé du gradient couleur.

2.7 Le gradient directionnel : utilisé pour mettre en évidence la présence des contours verticaux et horizontaux dans las images.

3 - Les méthodes de classification

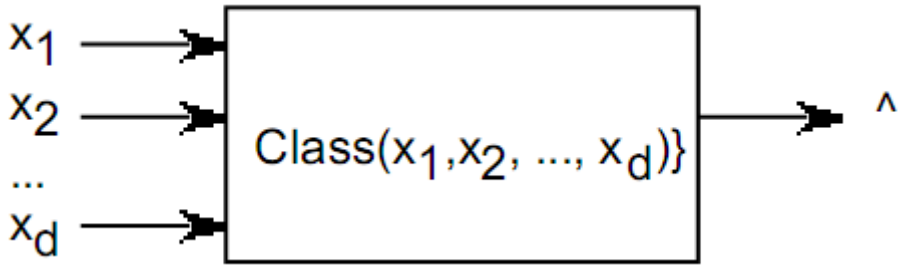
3.1 Classification Baysienne:

Plusieurs techniques d'apprentissages sont utilisées actuellement dans la classification des images [YAN02]. Une de ces techniques est la classification Bayesienne qui repose sur une

fonction de vérité probabiliste et la règle de Bayes. Soit les événements E décrit par un vecteur de caractéristiques X : (E,X).

Soit K classes d'événements {Tk} = {T1, T2, ..., TK}

La classification est un processus d'estimation de l'appartenance d'un événement à une des classes Tk fondée sur les caractéristiques de l'événement X. [FER06]



^k=Decider(E k)

La fonction de classification est composée de deux parties d() et gk():^k = d(g(X)).

g(X) : Une fonction de discrimination : RD RK

d() : Une fonction de décision : RK { K}

Dans un système de vérité probabiliste, la valeur de vérité de la proposition est une probabilité : p(k) = p(E k).

Le critère de décision est de minimiser le nombre d'erreur. Dans un système probabiliste, ça revient à minimiser la probabilité d'erreur. Ceci est équivalent à choisir la classe la plus probable. ^k = Decider(E k). = arg-max {p(k | X)}.

Pour estimer la probabilité nous utilisons les caractéristiques, X, de l'événement.

3.2 Les algorithmes K-means: La quantification vectorielle et la classification non-supervisée sont étudiés depuis plusieurs dizaines d'années [MAC67]. Les algorithmes de type "k-means" ont été développés parallèlement dans les deux domaines depuis les années cinquante. Les méthodes de quantification visent à représenter une source ("signal", "données") en un ensemble fini et réduit de symboles. Les techniques de classification sont utilisées dans le contexte de l'analyse de données pour le regroupement de données et exploitent les similarités dans l'espace de représentation de celles-ci. [FAU03]

*** Le seuillage degré 0 de la quantification:** soit {xj , j = 1, ...,N} l'ensemble de vecteurs de l'espace Rp qui est représenté par un nombre réduit de vecteurs {βi, i = 1, ...,C} du même

espace. La façon la plus simple est de définir les {βi} comme les sommets (ou les centres) des cellules d'une subdivision systématique de l'espace Rp et d'associer chaque xj à la cellule dans laquelle il se trouve. Autrement dit, il s'agit simplement de donner un seuil pour chaque composante réelle de chaque vecteur xj. Dans le cas de la description couleur d'images, le seuillage des valeurs de pixel est la technique la plus utilisée pour produire les histogrammes couleur.

*** k-means / GLA :** K-means et GLA ont le même principe, qui se résume de la façon suivante: l'algorithme suppose donnée une partition initiale. Chaque itération comprend trois étapes :

1- détermination de l'association entre les données et les classes,

2- mise à jour des centres de classes

3- mise à jour du critère de convergence.

Les itérations se déroulent jusqu'à convergence vers un optimum local, éventuellement global.

L'algorithme des k-means/GLA se déroule comme suit :

Soient les données {xj , j = 1, ...,N} et les prototypes {βi, i = 1, ...,C}.

1. A l'itération k=0, les prototypes {βi} sont initialisés.
2. Nouvelle itération k = k + 1.
3. Association aux classes : à tout prototype βi on associe la donnée xj la plus proche. Une classe Ci est définie comme : $\forall i \in \{1, ...,C\}$, $Ci = \{x \mid i = \text{argMin}_{i'}(d(x, \beta i0))\}$, où d(., .) est la distance entre données et prototypes.
4. Mise à jour des prototypes : chaque prototype βi pour l'itération suivante est défini comme le centroide de la classe Ci.
5. Calcul de l'erreur moyenne quadratique Ek de la partition.
6. Reprendre à l'étape 2 jusqu'à ce que $| Ek+1 - Ek |$ soit faible.

*** C-moyenne floues :** la classification floue introduit des degrés d'appartenance floue entre les données et les classes. Plutôt que de considérer que l'association, entre une donnée et une classe, est binaire, ce qui réduit le risque de la convergence du critère vers un optimum local non-global.

Les données sont autorisées à appartenir à plusieurs classes avec des degrés variables. Aux extrêmes, un degré nul exprime la non-appartenance à une classe et un degré de 1 signifie l'appartenance totale.

L'algorithme de classification floue le plus utilisé est le Fuzzy C-Means introduit par Bezdek en 1981. Il correspond littéralement à la version floue des k-means.

La partition optimale des données est obtenue par minimisation de la fonction objective suivante :

$$J = \sum_{i=1}^{C} \sum_{j=1}^{N} (u_{ij})^q d^2(x_j, \beta_i) \quad (25)$$

Sous la contrainte :

$$\sum_{i=1}^{C} u_{ij} = 1, \ i = 1, \ldots, N \quad (26)$$

Où uij ∈[0; 1] désigne le degré d'appartenance de la donnée xj à la classe de prototype βi.

q > 1 est le paramètre flou (plus q est grand, moins les valeurs d'appartenance sont marquées). La minimisation de cette fonctionnelle par rapport aux uij est obtenue, comme pour les k-means, itérativement. Elle conduit la convergence de la partition floue donnée par les uij.

L'algorithme C-moyenne floues:

1. A l'itération k=0, les prototypes {βi, i = 1, ...,C} et la matrice d'appartenance U=[uij] sont initialisés.
2. Nouvelle itération k = k + 1.
3. Calcul des degrés d'appartenance :

$$u_{ij} = \frac{\left[\frac{1}{d^2(x_j,\beta_i)}\right]^{1/(q-1)}}{\sum_{k=1}^{C}\left[\frac{1}{d^2(x_j,\beta_k)}\right]^{1/(q-1)}}, \forall i, j \quad (27)$$

4. mise à jours des prototypes :

$$\beta_i = \frac{\sum_{j=1}^{N} u_{ij}^q x_j}{\sum_{j=1}^{N} u_{ij}^q}, \forall i \quad (28)$$

5. Mise à jour du critère de convergence
6. Reprendre à l'étape 2 jusqu'à la stabilité des degrés d'appartenance (i.e. convergence vers zéro de $\| U^{(k+1)} - U^k \|$.

*** La classification K-PPV :** Cette méthode présentée dans [AME07]**,** non paramétrique n'exige pas de connaître la loi de distribution des variables. Pour un K fixé, ce classificateur fait voter les K plus proches voisins de x pour savoir à quelle classe x appartient. Ce classificateur est une extrapolation du classificateur euclidien [DEV82]. Au lieu d'utiliser le vecteur de caractéristique moyenne Mi comme unique prototype d'une classe, la méthode du

plus proche voisin fait intervenir tous les exemplaires des vecteurs caractéristiques disponibles. Pour ce faire, la distance euclidienne, entre chacun de ceux-ci et celui de l'objet à classifier, est calculée. La classe assignée à l'objet, est alors celle du prototype le plus proche de celui-ci.

Soit SN un ensemble de N paires de variables aléatoires et indépendantes :

$(\overline{X}1 , \theta1), (\overline{X}2 , \theta2),\ldots, (\overline{X}n , \theta n)$ provenant de c classes $\omega1, \omega2, \ldots., \omega c$ où θn est l'étiquette de la vraie classe de n x (avec n = 1,2,…N). Etant donnée une nouvelle variable X′ qui est indépendante de SN, on recherche dans SN les K plus proches voisins (K-PPV) Xk de X′ (avec i = 1,…,K), en écrivant que: $D(\overline{X} , \overline{X}ki) \leq D(\overline{X} , \overline{X}j)$

où i=1..k, et $j \neq \{K_{i , i=1..k}\}$ et D() est une distance définie dans l'espace des propriétés texturales est une distance définie dans l'espace des propriétés texturales. On affecte ensuite x à la classe ω_k en appliquant la règle suivante :

$$X \in wk \quad si \; N_{wk} = \underset{K'=1..c}{Max} \{N_{wk'}\} \tag{29}$$

Où :

$$N_{wk'} = \sum_{i=1}^{k} l (\theta_{ki} - w_{k'}) \tag{30}$$

avec :

$$l(X) = \begin{cases} 1 & si \; X = 0 \\ 0 & si \; X \neq 0 \end{cases} \tag{31}$$

3.3 Les réseaux de neurones : Les réseaux de neurones sont largement utilisés dans le domaine de classification des images. Dans cette cession nous faisons une brève définition du neurone formel, les réseaux récurrents et les réseaux non récurrents.

*** Neurone formel** S'inspirant des travaux scientifiques sur les neurones biologiques, McCulloch et Pitts (1943) ont proposé le modèle formel du neurone [SID02]. Le neurone formel fait une somme {w1 ,w2 ,...,wn} pondérée par des potentiels d'actions.

{e1 ,e2 ,...,en} qui lui parviennent. Chacun de ces potentiels {ei , i=1..n} et de ces poids {wi , i=1..n} est une valeur numérique qui représente respectivement l'état du neurone qui l'a émis et l'importance du lien avec ce dernier :

$S=\sum_i (e_i \times w_i)$ (32)

Le neurone formel s'active suivant la valeur S de cette sommation voir figure II.4:

- Si cette somme (S) dépasse un certain seuil (b) alors le neurone est activé et transmet une réponse (sous forme de potentiel d'action) dont la valeur (A) est celle de son activation (la fonction d'activation F du neurone). - Si le neurone n'est pas activé alors il ne transmet rien.

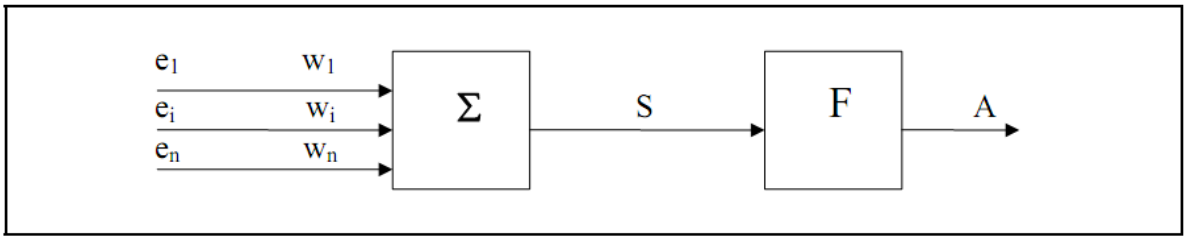


Figure II.4 Neurone formel avec fonction d'activation

La fonction d'activation F est une fonction à seuil. Mais, les chercheurs soucieux de conformité avec le modèle biologique donnent une description continue du neurone. La fonction F va produire un signal continu qui rend l'importance de l'activation du neurone. Cette fonction est généralement bornée, continue et croissante comme celle des fonctions sigmoïdes :

$$F1(x) = \frac{1}{1 + e^{-ax}} \quad (33) \quad \text{et} \quad F2(x) = \frac{1 - e^{-ax}}{1 + e^{-ax}} \quad (34)$$

qui sont respectivement à valeurs dans [0,1] et dans [-1,+1]. Le paramètre (a) est le gain qui mesure la raideur de la sigmoïde.

*** Réseau neuronal :** Un réseau neuronal est un ensemble de neurones formels interconnectés et évoluant dans le temps par interactions réciproques. La description statique de la fonction d'activation d'un neurone formel n'est pas suffisante pour caractériser un réseau.

Ainsi, un réseau neuronal se définit par :

- son architecture, qui représente la structure de ses connexions,
- les fonctions d'activation de ses neurones,
- la dynamique de ses connexions.

La structure des connexions dans un réseau neuronal peut aller d'une connectivité totale (les neurones sont totalement connectés entre eux) à une connectivité locale. Les poids des connexions sont des valeurs qui leur sont attribuées afin de rendre compte de l'influence d'un neurone sur ceux qui sont reliés à sa sortie.

Les nombreux modèles de réseaux neuronaux forment deux grandes familles :

- les réseaux non récurrents,
- les réseaux récurrents.

*** Les réseaux non récurrents :** La particularité du réseau non récurrent est qu'il est organisé en couches successives. Le premier modèle opérationnel a été le « Perceptron de Rosenblatt » dans les années 1950. Dans ce réseau, les connexions ne vont que dans un seul sens de la couche d'entrée vers la couche de sortie. Par ailleurs, il n'y a pas de connexion entre les cellules d'une même couche.

Chaque connexion entre les cellules d'association et les cellules de décision est affectée d'un poids. L'apprentissage du « Perceptron » est basé sur la règle de Hebb, c'est-à-dire un apprentissage supervisé qui se fait par correction d'erreur.

Cet apprentissage consiste à présenter au réseau une série d'exemples ou couples (E,S) où E est l'entrée et S est la sortie désirée, puis à minimiser l'erreur entre la sortie désirée S et la sortie effective Y. L'algorithme est décrit comme suit :

1. Réseau = ensemble de neurones, Pour le neurone j : {ei , i=1..n} et Sj ,
2. Calculer les sorties obtenues: pour le neurone j , $Yj = \sum i \neq j \ (ei * \omega i)$
3. Calculer les nouveaux poids des connexions : pour la connexion entre les neurones i et j ,

   $\omega ij \ (t+1) = \omega ij(t) + a(Sj-Yj)$

   ωij : le poids de la connexion entre i et j

   Sj : la sortie désirée pour j

   Yj : la sortie obtenue (effective) pour j

   a : coefficient de la vitesse d'apprentissage

   t : le temps

   i,j : neurone i et neurone j

*** les réseaux récurrents:** Le réseau est dit récurrent s'il existe un circuit dans son graphe orienté de connexion. Il existe deux modèles de réseau récurrent :

- le modèle de Hopfield : réseau entièrement connecté
- le modèle de Kohonen : réseau partiellement connecté.

3.4 Le classificateur à distance minimum : Ce classificateur est non paramétrique qui permet de construire les frontières des classes à partir du barycentre et de la matrice de variance - covariance de chaque classe. Pour une distance donnée, ce classificateur permet d'avoir une indication sur la compacité et la forme de la distribution des caractéristiques texturales dans la classe par rapport à leur barycentre. La structure du classificateur à distance minimum est globalement la suivante:

Soit N échantillons $X_1, X_2, .. X_n$ issus de c classes $\omega_1, \omega_2, \ldots, \omega_c$

le barycentre de chaque classe est calculé en faisant:

$$\overline{Z_k} = \frac{1}{N_k} \sum_{\overline{x_k} \in w_k} \overline{X_k} \quad (35)$$

où k = 1 ,…., c et Nk est l'effectif de la classe ωk. Etant donnée une nouvelle observation x, on affecte x à la classe ωk en appliquant la règle suivante :

$$\overline{X} \in w_k \quad \text{si } D(\overline{X}, \overline{Z_k}) = \min_{K'=1..c} D(\overline{X}, \overline{Z_k}') \qquad (36)$$

où D(.) est une distance entre deux vecteurs.

3.5 Classification hiérarchique: après l'extraction des zones homogènes dans l'image, une méthode de classification automatique s'avère nécessaire. Celle-ci a pour but de partitionner l'image multi-spectrale en classes disjointes. Il existe deux types de méthodes de classification hiérarchique, nous citons:

- La méthode hiérarchique descendante,
- La méthode hiérarchique ascendante [HAR82].

Quant aux méthodes hiérarchiques ascendantes. Celles-ci procèdent par des groupements successifs. Initialement, le nombre de pixels reflète le nombre de classes. Les pixels les plus similaires sont par la suite groupés jusqu'au nombre de classes choisies initialement. Le groupement des pixels se base sur le critère de la distance euclidienne. Pour une classification hiérarchique ascendante d'un ensemble de points, nous devons :

1- Disposer d'un ensemble E de b éléments à classifier où b est le nombre de pixels homogènes à traiter,

2- Chercher les deux éléments les plus proches que l'on agrège en un nouvel élément,

3- Calculer les distances entre le nouvel élément et les éléments restants. On se trouve dans les mêmes conditions qu'à l'étape 1, avec seulement (b - 1) éléments à classer,

3.6 Algorithme de KRUSKAL: soit E un ensemble à classifier constitué de b pixels homogènes, D une matrice de distance obtenue à partir de l'ensemble E, A est un ensemble vide. L'algorithme est défini par les étapes suivantes :

1 - Ranger dans l'ordre croissant l'ensemble des distances euclidiennes $d(E_i,E_j)$, calculées précédemment. Une arête E_iE_j est associée à chaque distance $d(E_i,E_j)$,

2 - Mettre dans l'ensemble A, la première arête de la liste,

3 - Mettre dans l'ensemble A l'arête suivante sauf si un cycle peut être formé avec les arêtes qui sont déjà dans A

4 - Arrêter le processus si le graphe (E, A) est connexe, sinon recommencer la troisième étape. [OUA99]

4 Graphe de comparaison appliqué pour la recherche d'image: Cette technique est appliquée dans la comparaison des images. Chaque niveau dans le graphe représente la dissimilarité entre l'image recherchée est l'image source.

Si la dissimilarité est inférieure à un seuil $\Im$ les nœuds seront positionnés au niveau le plus faible.

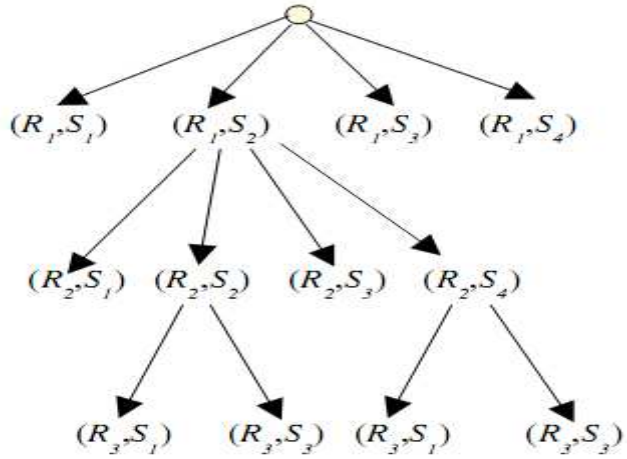


Figure II.5 Arbre de recherche, chaque nœud correspond à une comparaison entre les régions R et S

La dissimilarité entre deux graphe est calculée en parcourant chacun depuis le sommet jusqu'au nœud feuille et en ajoutant à chaque fois la valeur de dissimilarité correspondante au nœud. Ensuite toutes les dissimilarités sont rangées par ordre décroissant et celles qui sont supérieures au seuil $\Im$ seront rejetées par contre le reste des nœuds seront rangés dans la liste L. par exemple la liste L obtenue selon le graphe (figure II.5) est :

L = {(R_1, S_3), (R_2, S_3), (R_1, S_4), (R_3, S_3), (R_1, S_1), (R_2, S_1),(R_2, S_4), (R_3, S_1), (R_2, S_2), (R_1, S_2)}

Chaque chemin depuis la racine vers le nœud feuille correspond à une comparaison acceptée entre deux régions, par exemple dans le cas précédant**,** aussi chaque région cible peut être comparée avec plusieurs régions sources.

Distance entre graphes : soit M={ (R_1, S_1), (R_2, S_2), … , (R_m, S_m)} où R_i est la région source est Si est la région cible, soit d la distance entre régions. La fonction de dissimilarité est donnée par :

$$F(M) = \frac{1}{m} \sum_{i=1}^{m} d(R_i , S_i) \qquad (37)$$

Enfin, la distance entre deux images est F(M) minimum calculé.

5 Structure hiérarchique pour l'indexation des images:

Chaque image peut être représentée par sa couleur, sa forme, sa texture… etc. Ce qui augmente considérablement la taille des vecteurs caractéristiques. Pour indexer des documents images dans une large base documentaire, une représentation sous forme d'arbre sera très utile à cause de sa forme hiérarchique qui permet d'effectuer une recherche très rapide [SHA08].

Algorithme de base : l'algorithme de recherche dans telle structure est le suivant :

1- l'utilisateur extrait les caractéristiques de l'image et les stocke dans un vecteur f_{query}
2- l'utilisateur cherche le nœud racine depuis la base de données
3- en utilisant f_{query} et le nœud racine l'utilisateur décide soit de prendre le chemin droite (sous arbre droit) ou le chemin gauche (sous arbre gauche).
4- Pour récupérer les données du nœud i l'utilisateur envoie une requête Qi sachant que le nœud racine est du niveau 0
5- La base de données interrogée retourne le résultat Ai pour la requête Qi
6- L'utilisateur utilise la fonction f(Ai) afin de savoir si le nœud Ai est une feuille, si ce n'est pas le cas (nœud racine) aller à l'étape 3

Nous pouvons, ainsi, utiliser des arbres de recherche binaires. Dans telles arbre chaque niveau i contient 2^i nœuds, touts les nœuds d'un niveau donné peuvent être représentés par un tableau. Le nœud recherché peut être un des 2^i nœuds.

La recherche dans des structures binaires peut être améliorée en faisant des calculs au niveau de l'utilisateur final. Ces calculs consistent à mesurer la distance entre les vecteurs caractéristiques à chaque niveau. Une petite amélioration de l'algorithme de base peut en simplifier les choses:

1. l'utilisateur extrait le vecteur caractéristique Fq à partir de la requête
2. à partir des informations sur le nœud racine, l'utilisateur décide du prochain chemin.
3. si le nœud recherché est dans le niveau i, l'utilisateur établi une requête Qi de recherche sur le nombre de nœud dans le niveau i.
4. la base de données reçoit la requête Qi et renvoie la réponse Ai.
5. l'utilisateur reçoit Ai et récupère les informations sur le nœud recherché.

6. si le nœud trouvé est une feuille, l'utilisateur récupère les résultats sinon il fait appel à la stratégie de recherche pour accéder au nœud suivant.
7. pour chaque nœud obtenu dans l'étape 6 on applique récursivement le même algorithme à partir de l'étape 3.

6 Système d'indexation en sémantique latente LSI: ce système est basé sur l'adaptation de la technique d'indexation en sémantique latente [ELG05]. LSI sert à diminuer la dimension d'une matrice A, qui représente un ensemble de documents, en utilisant une décomposition en valeurs singulières SVD, $A=USV^T$ où :

U est une matrice qui décrit les termes, de taille M x N $U.U^T = I_M$

V est une matrice de description des documents, de taille NxN $VV^T = I_N$

S est une matricer diagonale de N x N

Le problème consiste à représenter la fréquence d'apparition des mots clés par des valeurs numériques

A chaque attribut sont associés des paramètres qui varient dans une intervalle [min,max], le schéma suivant illustre le processus d'adaptation:

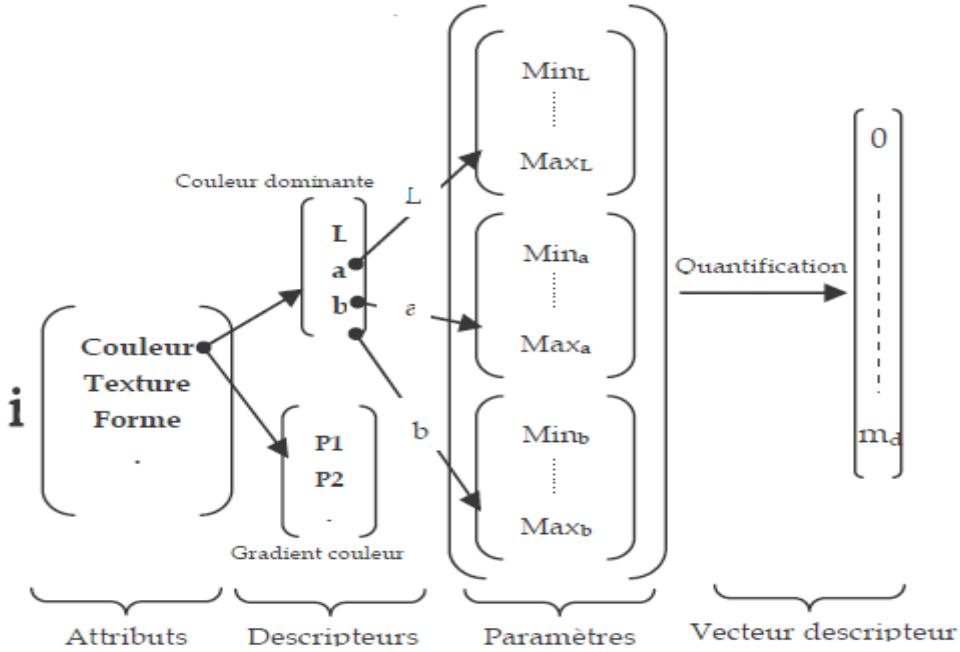


Figure II.6 : Approche d'adaptation [ELG05]

Donc, à chaque image on associe des descripteurs qui sont à leurs tours composés d'un ensemble de paramètres qui constituent le vecteur caractéristique.

La méthode LSI utilise un nombre important de descripteurs afin de combler le manque de la prise en compte de la sémantique dans la recherche des images parmi ces descripteurs on a : les couleurs dominantes et le gradient directionnel

7 Conclusion : Dans ce chapitre nous avons vu quelques caractéristiques des images ainsi que les systèmes, les plus connus, de classification et d'accès aux images par leurs contenus. Les systèmes de classifications sont assez nombreux ce qui fait que chaque système présente certains avantages par rapport aux autres pour certaines caractéristiques des images.

1. Introduction

Dans le deuxième chapitre nous avons présenté un aperçu global de la recherche et d'indexation des images de tout type. Dans ce chapitre nous allons aborder les spécificités et particularités des techniques d'indexation et de recherche d'images des anciens manuscrits.

2. Approches d'indexation des documents anciens

Il existe deux approches pour indexer des documents anciens. L'une est dite globale puisqu'elle exploite des caractéristiques globales et l'autre analytique ou locale qui emploie des descripteurs extraits localement.

2.1. Approches globales ou holistiques

2.1.1. Word spotting

Cette approche proposée par Tony M. Rath · R. Manmatha [RAT07] à été appliquée sur des documents anciens de l'université Isaac Newton à Washington, le document est considéré comme un ensemble des images de mots, le document est segmenté en image de mot puis le système de classification sert à calculer les distances entre les images et ceux qui se

ressemblent seront regroupées en clusters, chaque cluster sera labellisé manuellement et sert d'indexe vers le document approprié.

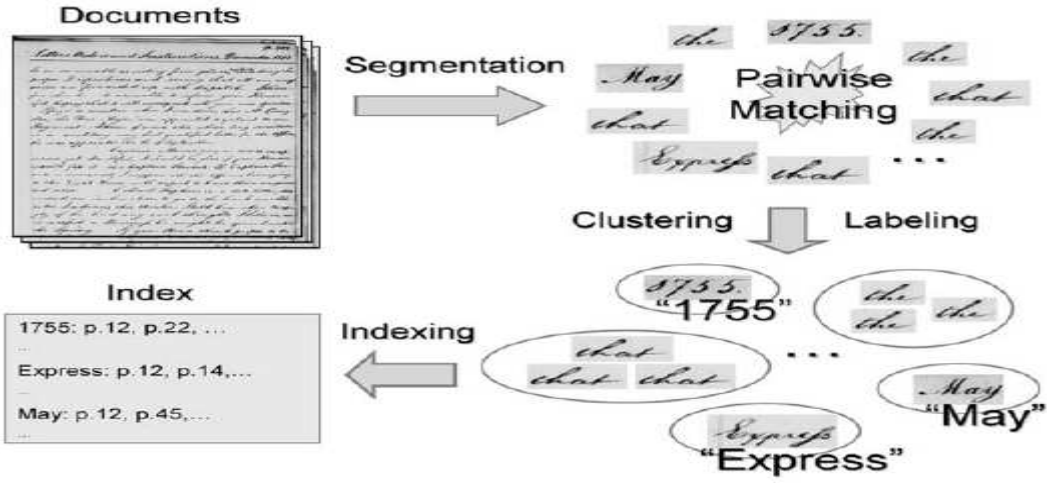


Figure III.1 schéma d'indexation par "word spotting"

La sélection des indexes se fait automatiquement car la sélection manuelle est très coûteuse. Pour effectuer un choix automatique des indexes le système utilise le graphe de fréquences des mots. Les mots significatifs qui peuvent servir d'indexes sont ceux qui se trouvent au milieu du graphe (fréquence moyenne). Les mots qui ont une haute fréquence sont souvent les mots de liaison tels que : ou, et,…etc. Les mots qui ont une faible fréquence sont des mots rarement rencontrés dans le texte, ces deux dernières catégories ne peuvent pas servir d'indexes.

*** Extraction des caractéristiques :** Les images traitée sont en niveau de gris avec 256 niveau [0..255] l'intensité des pixels I est donnée par I(r,c) où r et c indiquent la ligne et la colonne du pixel. Les caractéristiques retenues sont les suivants :

1- Projection du profil : cette caractéristique vise à calculer la distribution des pixels dans l'image. Elle est calculée par :

$$pp(I,c) = \sum_{r=1}^{h} (255 - I(r,c)) \qquad (38)$$

2- Profil des mots : le système calcule le profile du mot inférieur et supérieur qui sont donnés par :

$$up(I,c) = \begin{cases} \text{indéfinie} & \text{si } \forall\, r \text{ is\_ink}(I,r,c) = 0 \\ \end{cases} \qquad (39)$$

$$lp(I,c) = \begin{cases} \text{indéfinie} & \text{si } \forall\, r \text{ is\_ink}(I,r,c) = 0 \\ \underset{r=1..n}{\text{argmin}}\,(\text{is\_ink}(I,r,c) = 1) & \text{autrement} \\ \underset{r=1..n}{\text{argmax}}\,(\text{is\_ink}(I,r,c) = 1) & \text{autrement} \end{cases} \quad (40)$$

is_ink(I, r, c) est une fonction qui retourne 1 si le pixel est noir, sinon elle retourne 0

3- Transition encre/fond : La fonction nbit(I, c) calcule le nombre de transition encre/fond. Le paramètre défini est donné par : f4(I, c) = nbit(I, c)/6. [LEE02]

*** comparaison des mots:**

La comparaison des mots se fait à l'aide de l'algorithme DTW dont voici l'algorithme :

soit X=(x1,…,xm), Y= (y1,…,yn), Fonction distance d(.,.) , matrice D

l'algorithme DTW est le suivant :

1- D(1,1)=d(x1,y1) ;
2- For m=1 : M
3- D(m,1) = D(m-1,1) + d(xm,y1) ;
4- For n=1 : N
5- D(1,n) = D(1,n-1) + d(x1,yn) ;
6- For m=2 : M
7- For n=2 : N

$$8\text{- } D(m,n) = \begin{Bmatrix} D(m,n-1) \\ D(m-1,n) \\ D(m-1,n-1) \end{Bmatrix} + d(xm,yn)$$

2.1.2 Word Matching: Cette méthode proposée dans [ADA07] est basée sur la comparaison des mots en utilisant leurs contours fermés, les auteurs ont préférés la comparaison des contours au lieu des mots complets car cette proche peut éliminer l'étape de correction des inclinaisons.

Pour effectuer la détection des contours l'approche utilise une segmentation multi échelles qui consiste à fournir un détail du contour à plusieurs niveaux.

L’extraction de contour se fait en cinq étapes : binarisation, estimation de la position, étiquetage des composants connectés, attachement des composants déconnectés et traçage du contour.

1- Binarisation : Dans cette phase les pixels sont séparés en deux : pixels appartiennent au teste et pixels appartiennent au fonds, ainsi le seuillage utilisé est un seuillage dynamique à cause des variations considérables du contraste dans les documents anciens, le seuil est évalué dans chaque pixel et est donné par : $T=\mu(1- k(1- \sigma/R)$) où , σ est la l’écart type du niveau de gris dans une petite fenêtre au voisinage du pixel, k est une constante fixée à 0.02 et R= 128. Cette formule est proposée par Sauvola et Al [SAU97]. Voir figure III.2

2-

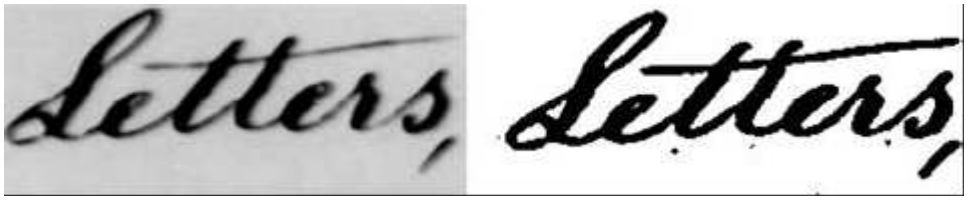


Figure III.2 Binarisaion [ADA07]

2- Estimation de la position: La binarisation, en générale, ne donne pas un contour fermé ce qui a poussé à faire d’autres gymnastiques afin d’avoir un contour continu. Pour ce faire ils ont commencé par une estimation de la ligne de base de chaque mot, cette ligne est extraite en calculant le nombre de pixels dans chaque ligne, elle est notée par *x*-height

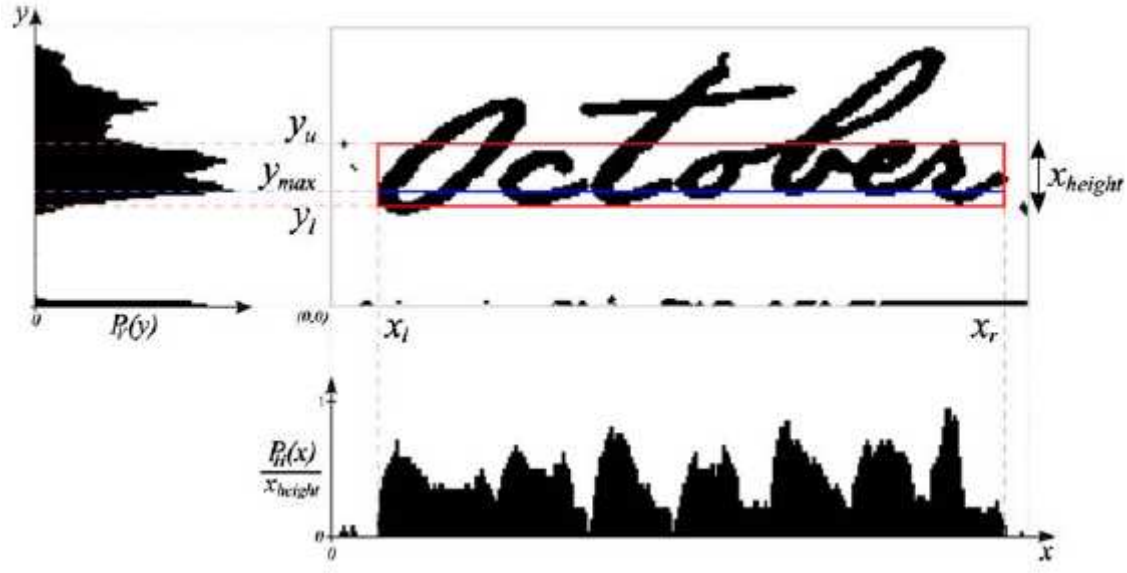


Figure III3 estimation de la position

3- étiquetage des composants connectés : l'étiquetage des pixels continu consiste à coder les pixels noirs connectés à partir de l'image binaire par 8 voisinages.

4- Attachement des pixels déconnectés: la déconnexion des composants est rarement due aux seuils dynamiques utilisés durant la phase de binarisation. En générale, on la constate entre les lettres, un lien est considéré valide s'il vérifie les conditions suivantes:

* le lien est à l'intérieur du carré englobant la lettre minuscule, la coordonnée verticale du lien, y vérifie la condition $yl < y < (yu - \alpha_c x height)$ où α_c est fixée à 0.02 figure précédente

* les deux limites du lien sont au dessus du carré englobant la lettre minuscule, la coordonnée verticale du lien, y vérifie la condition $y > 2y_u - y_l$. Voir figure III.4

Figure III.4 Résultats d'extraction de contour

5- Traçage de contour : après la connexion des composants séparés on procède au traçage du contour. Voir figure III.4.

* Comparaison des contours : Pour comparer les images le système effectue une comparaison de leurs contours, cette comparaison sera basée sur un algorithme simple de comparaison de forme, le système a introduit, également, une représentation multi échelles concavité/convexité 'MCC' au différents niveau relevés dans chaque contour. Une représentation 'MCC' peut être représentée comme suit : Soit M une matrice où les lignes représentent les niveaux d'échelle σ et les colonnes représentent les points de contour u.

$M(\sigma,u)$ contient l'information sur le degré de concavité/convexité du $u^{ème}$ point contour à l'échelle σ voir figue III.5

Avant de procéder au calcul de la représentation MCC, le contour subit un amincissement à l'aide du filtre Gaussien/Kernel suivant :

$$X\sigma(u) = \int x(t)\, \phi\sigma(u - t)\, dt \qquad (41)$$

$$\phi\sigma(u) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-u^2/2\sigma^2} \qquad (42)$$

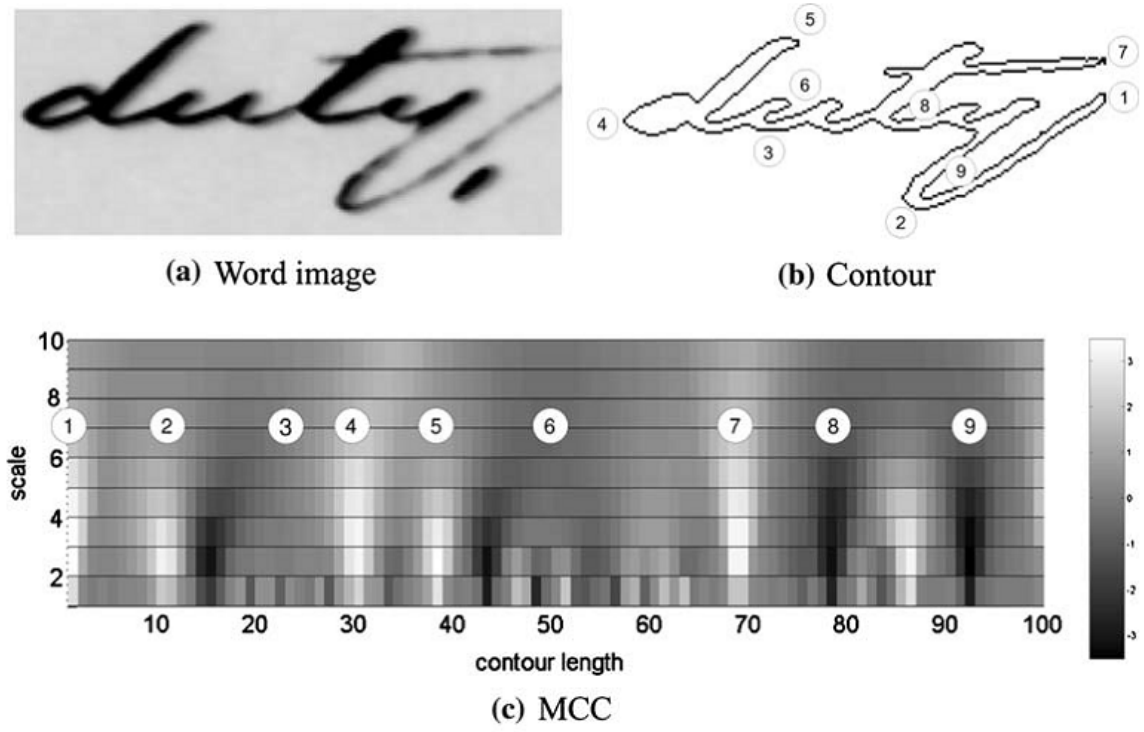


Figure III.5 Extraction de la représentation 'MCC' [ADA07]

2.1.3 Comparaison des mots dans les documents imprimés en utilisant les données synthétiques : cette méthode a été proposée par T. Konidaris [KON07] elle traite les documents grecs imprimés de la période 1471–1821. Elle est ainsi basée sur les points suivants voir figure III.6:

1-Création des images des mots

2-segmentation des mots

3- extraction des caractéristiques pour chaque mot

4- la recherche qui est renforcée par une interaction avec l'utilisateur

Le schéma suivant illustre le déroulement des étapes d'indexation et de recherche :

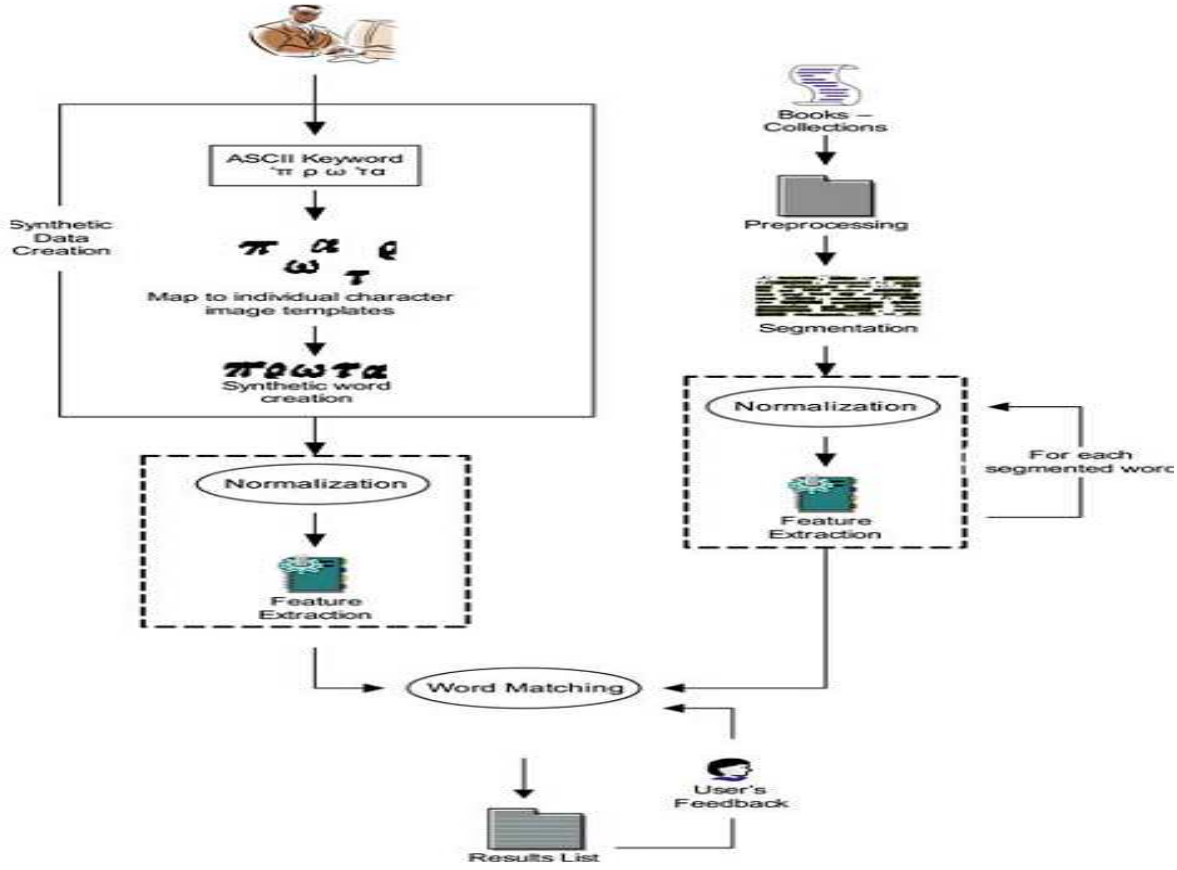


Figure III.6 architecture du système [KON07]

* Prétraitement : dans cette phase le système commence par une binarisation pour isoler l'écriture du fond, le seuil sera définit après une estimation des pixels noir, puis une combinaison avec l'image originale et enfin une phase de traitement postérieur consiste à connecter les trait séparés du même mot.

* Segmentation : Dans cette phase le texte est divisé en mots, l'algorithme utilisé est RSLA (Run Length Smoothing Algorithm), basé sur des paramètres dynamiques dépendants de la taille des caractères.

* Création de données synthétiques: premièrement l'utilisateur tape le mot clé, le système identifie l'image à partir du code ASCII puis ajuste l'ensemble de caractères sélectionnés sur la ligne de base. Voir figure III.7

keyword	Synthetic data without baseline adjustment	Synthetic data with baseline adjustment
σωματα		

Figure III.7 création de données synthétiques à partir de la représentation ASCII

* Extraction des caractéristiques : l'extraction des caractéristiques se fait en deux phases : normalisation et extraction des caractéristiques hybrides. La normalisation est le processus qui ajuste la longueur et la largeur des mots. Pour extraire les caractéristiques hybrides, premièrement l'image du mot est divisée en morceau et la densité des pixels noirs dans chaque zone sera calculée comme suit :

Soit im(x,y) une fonction qui retourne 1 si le pixel appartient au fond et 0 si le pixel appartient au texte, Xmax, et Ymax sont respectivement la largeur et la longueur du mot, Zh et Zv sont respectivement le nombre total de zones horizontales et verticales.

Les caractéristiques $f\,z(i)$, $i = 0, \ldots, ZHZV - 1$ sont calculées comme suit :

$$f^z(i) = \sum_{x=x_s(i)}^{x_e(i)} \sum_{y=y_s(i)}^{y_e(i)} \mathrm{im}(x,y) \qquad (43)$$

Où :

$$x_s(i) = \left(i - \left\lfloor \frac{i}{Z_H} \right\rfloor Z_H\right) \frac{x_{max}}{Z_H} \qquad (44)$$

$$x_e(i) = \left(i - \left\lfloor \frac{i}{Z_H} \right\rfloor Z_H + 1\right) \frac{x_{max}}{Z_H} \qquad (45)$$

$$y_s(i) = \left\lfloor \frac{i}{Z_H} \right\rfloor \frac{y_{max}}{Z_V} \qquad (46)$$

$$y_e(i) = \left(\left\lfloor \frac{i}{Z_H} \right\rfloor + 1\right) \frac{y_{max}}{Z_V} \qquad (47)$$

Deuxièmement, les projections du profil des mots sont calculées; chaque mot sera divisé en deux sections séparée par une ligne horizontale y = yt qui traverse le centre de l'image du mot tel que:

$$y_t = \frac{\sum_x \sum_y \mathrm{im}(x,y) \cdot y}{\sum_x \sum_y \mathrm{im}(x,y)} \qquad (48)$$

Le profil supérieur respectivement inférieur sera défini en calculant la distance entre la ligne horizontale y=yt et le pixel le plus haut, respectivement le plus bas comme le montre la figure III.8

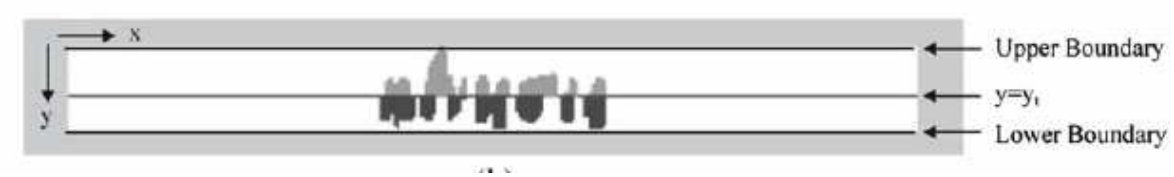


Figure III.8 Caractéristiques indiquant les profils supérieur et inférieur

$$y_{\text{up}}(x) = y_t - y_0,$$

$$\text{where } y_0 = \begin{cases} y_t, \text{ if } \sum_{y=0}^{y_t} \text{im}(x,y) = 0, \\ y: \ (\text{im}(x,y) = 1 \ \& \ y = \min(y_i)), \\ y_i \in [0, y_t], \qquad \text{otherwise} \end{cases} \tag{49}$$

$$y_{lo}(x) = y_0 - y_t,$$

$$\text{where } y_0 = \begin{cases} y_t, \text{ if } \sum_{y=y_t}^{y_{\max}} \text{im}(x,y) = 0, \\ y: \ (\text{im}(x,y) = 1 \ \& \ y = \max(y_i)), \\ y_i \in [y_t, y_{\max}], \qquad \text{otherwise} \end{cases} \tag{50}$$

$$f^p_{\text{upper\_area}}(i) = \sum_{x=x_s(i)}^{x_e(i)} y_{\text{up}}(x) \tag{51}$$

$$f^p_{\text{lower\_area}}(i) = \sum_{x=x_s(i)}^{x_e(i)} y_{\text{lo}}(x) \tag{52}$$

Où :

$$x_s(i) = \left(i - \left\lfloor \frac{i}{P_V} \right\rfloor P_V\right) \frac{x_{\max}}{P_V} \tag{53}$$

$$x_e(i) = \left(i - \left\lfloor \frac{i}{P_V} \right\rfloor P_V + 1\right) \frac{x_{\max}}{P_V} \tag{54}$$

- Recherche de l'image : la recherche consiste à comparer entre le mot introduit par l'utilisateur et les mots contenus dans la base d'indexes. La distance L1 entre les deux mots est donnée par :

  Dist (f_q(i) , f_{db}(i)) = $\sum_i$ || f_q(i) – f_{db}(i) || (55)

Où : $f_q(i)$ concerne le mot introduit par l'utilisateur et $f_{db}(i)$ concerne le mot de la base d'indexes.

Après cette phase de calcul, le système propose une liste d'index qui sont plus proches au mot recherché, ensuite l'utilisateur intervient pour sélectionner les index les plus pertinents. Voir figure III.9

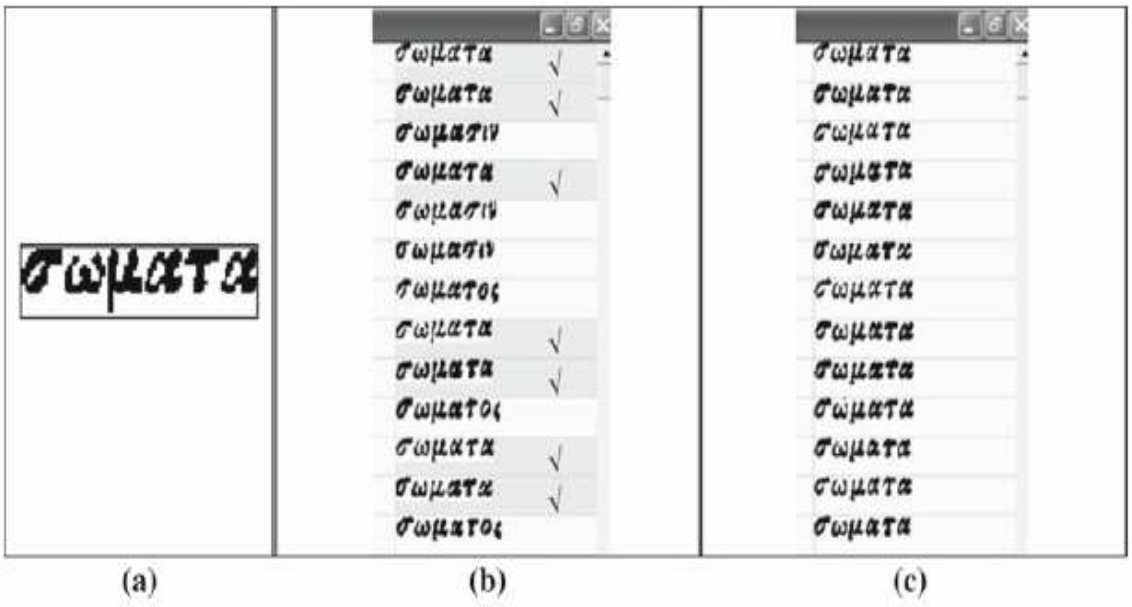


Figure III.9 recherche du mot "*σωματα*"

2.1.4 Indexation de documents anciens par la forme de mots:

Ce travail à été réalisé par J. Llados et G. Sanchez dans leur approche [LLA07]. La base de cette approche est de calculer le contexte de forme à partir des points constituants le squelette de l'image, et de le considérer comme un descripteur de base de cette image.

Les signatures de forme de mot: la signature d'une forme est le contexte de description de cette forme, elle est définie comme suit : soit p={p1,p2….pn} un ensemble de n points caractéristiques de la forme, le contexte de forme en un point pi est défini par l'histogramme hi des coordonnées relatives aux autres points autour de pi [BEL02].

Donc $hi(k) = \#\{q \neq pi : (q - pi) \in zone(k)\}$ (56)

L'espace autour de pi est divisé en zones homogènes, et hi(k) représente les densités des points dans la $k^{ème}$ zone. Voir figure III.10

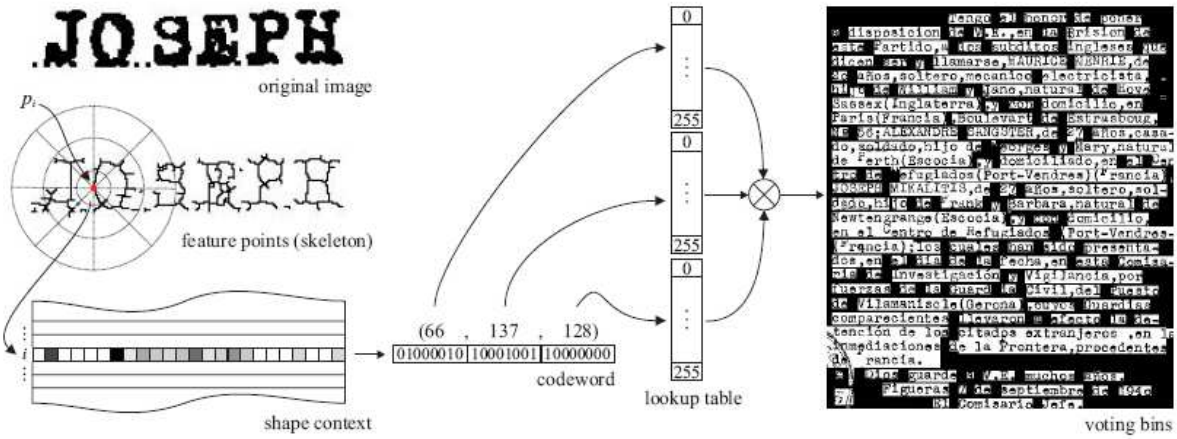


Figure III.10 signature de forme basée sur son contexte [LLA07]

* Création des indexes : les indexes sont rangés dans un tableau afin de mieux gérer la relation mots clés - documents, et de faciliter aussi la recherche. Chaque image est divisée en zones et chaque zone sera indexée par le même mot clé. Donc pour chaque zone W le squelette et les mots clés sont calculés, soit W={b1,b2,..bn} où n est le nombre de points squelette de W. pour chaque mot clé *bi* = (*β*1i, *β*2*i* , *β*3*i*) une valeur (I,W) sera insérée dans le tableau comparatif, voir figure III.10.

2.1.5 Système d'indexation d'image de lettrines :

Ce système proposé dans [DEL06] est un système d'indexation des parties graphiques. Plus particulièrement, appliqué à l'indexation des images de lettrines. Le but de cette approche est le traitement de larges bases d'images.

Les travaux sont centrés d'avantage sur les aspects complexité que sur la construction d'index hautement discriminant. L'objectif est alors d'indexer les images de lettrines appartenants à de larges bases. Pour ce faire, on a comme cas d'usage d'indexation la recherche d'images de lettrines de même classe de tampon. La Figure III.11 en donne quelques exemples. Chacune de ces images correspond à la numérisation d'une impression papier produite par un tampon. On qualifie usuellement les impressions papier de ces images d'estampes ou d'empreintes. Les tampons étaient en effet utilisés pour produire différentes estampes sur un même ouvrage, ou entre ouvrages différents. Ils pouvaient également être dupliqués en cas d'usure ou afin d'accélérer les processus d'impression.

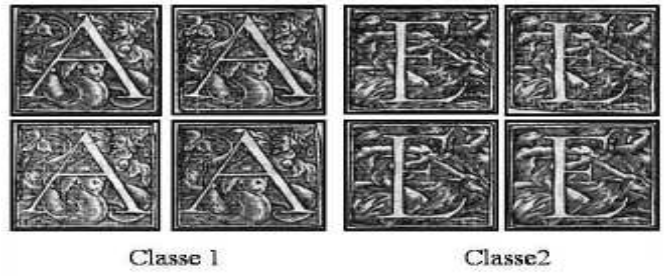


Figure III.11 Exemple de classe de tampon

La figure suivante détaille l'architecture du système :

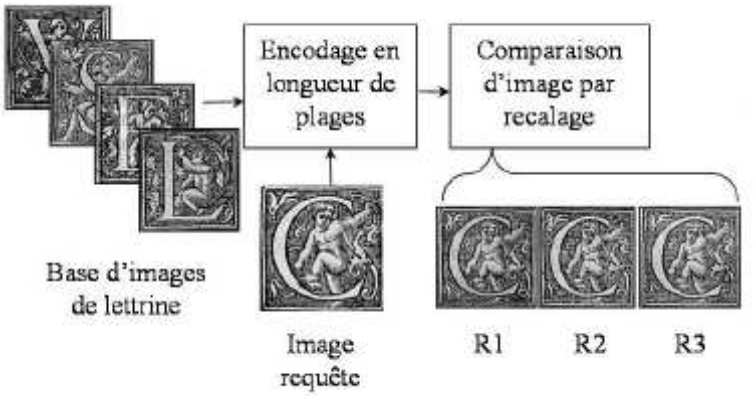


Figure III.12 Architecture du système [DEL06]

* Encodage en longueur de plage : le système utilise deux approches, la première consiste à utiliser des architectures matérielles dédiées, comme les processeurs pipelines, les ordinateurs massivement parallèles…etc. Cependant, la montée en puissance des machines et le développement des réseaux et d'Internet dans les années 90 ont quelque peu fait tomber en désuétude cette approche. L'autre approche consiste à employer des représentations adaptées aux algorithmes utilisés dans des buts de réduction de complexité. Il existe peu de travaux traitant de cette thématique. Une méthode utilise une représentation basée sur les contours pour exécuter rapidement des algorithmes exploitant des masques de voisinage comme l'érosion, la squelettisation …etc. Une autre propose un système pour la détection rapide de contour basé sur une représentation à base de plages. Enfin, les auteurs dans [BIA 96] utilisent une représentation à base de composantes connexes pour la recherche rapide de sous-structures au sein d'images de documents. De cette manière, tous ces systèmes exploitent des représentations adaptées aux algorithmes qu'ils mettent en oeuvre. Dans ce système [DEL06] les auteurs s'intéressent à la représentation à base de plages. La plage permet en effet de représenter de façon compacte des successions de pixels. Celles-ci sont généralement appliquées aux images binaires, la Figure III.13 en donne un exemple.

Une plage est une séquence maximale de pixel définie par un triplet {o,(x, y),l} où : o est l'orientation de la plage (verticale ou horizontale) (x,y) est le point d'origine de la plage et l la longueur de la plage en pixel

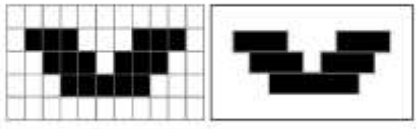


Figure III.13 Exemple d'encodage en longueur de plage

* Comparaison d'images par recalage : l'algorithme de comparaison procède en deux étapes : une étape de recalage suivie d'une étape de calcul de distance. Le recalage des images est basé sur l'analyse de leurs histogrammes des projections verticales et horizontales. La Figure III.14 en donne des exemples. En effet, la segmentation des images de page de livres anciens imprimés induit l'apparition de couleur du fond de dimensions variables sur les images de lettrines. Les paramètres de recalage se déduisent donc intuitivement à partir de l'analyse des pixels de couleur de la forme. Les histogrammes sont construits à partir des parcours des plages verticales et horizontales des images. La comparaison se fait deux à deux pour chaque couple histogrammes (soit horizontaux soit verticaux). Pour ce faire, une distance pondérée est calculée entre couple d'histogrammes {g, h} . La pondération rend en effet la comparaison des histogrammes moins sensible aux fortes variations d'amplitudes.
Un calcul de distance par pondération semble donc plus adapté aux histogrammes des images de lettrines qui présentent ces fortes variations. Les images pouvant être de dimensions différentes, une distance pondérée est calculée par offset {0, ., l − k}. Le delta (en x ou y) à appliquer correspond alors à l'offset de distance minimum. La Figure III.14 donne deux images présentant un recalage de dx = 1 et dy = 4.

$$\text{Delta} = \min \left(\bigcup_{j=0}^{l-k} \square \; \frac{|\,(h_i - g_{i+j})\,|}{h_i} \right) \qquad (57)$$

où: g=1,2,…,k , h=1,2,…,l , et $k \leq l$

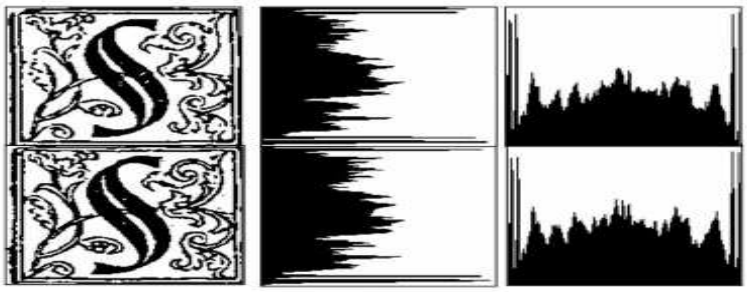


Figure III.14 Comparaison d'histogramme de comparaison

L'algorithme utilisé pour la comparaison est le suivant:

```
S =0,    x1 =0,   x2 = 0 , a1 = 0, a2 =0
Pour chaque ligne  L1 d'indice y de i1
    et L2 d'indice y + dy de i2
    faire
        p1 := suivant (L1)  x1+= p1.long
        p2 := suivant (L2)  x2+= p2.long
      tant que (p1<> fin) ou (p2  <> fin)
              tant que (x1 ≥ x2+dx)
                si p2.couleur = p1.couleur
                      alors
                      x2+=p2.long
                      s+=p2.long – a2
                      a2 = 0
                 fin
               fin
             tant que (x1 ≤ x2+dx)
                si p2.couleur = p1.couleur
                      alors
                      x1+=p1.long
                      s+=p1.long – a1
                      a1 = 0
                   fin
               fin
      fin
s=s / min(i1.largeur, i2.largeur) x i1.hauteur
```

Cet algorithme de comparaison d'images procède donc en deux étapes successives de recalage puis de calcul de distance. Il utilise pour cela les images encodées et préalablement stockées. Il parcourt ces images encodées à différentes reprises afin de les comparer. Tout d'abord, deux parcours sont nécessaires par image pour l'extraction des histogrammes verticaux et horizontaux. Cependant, ces histogrammes étant des méta-données naturelles de la représentation à base de plages sont calculés et stockés préalablement à l'encodage. Par la suite deux autres parcours sont utilisés pour le calcul de la distance. Ces parcours sont effectués à la fois sur les images cible et requête.

2.1.6 Filtre d'HERMITE et GABOR pour la classification des documents anciens:

Dans [EGL07], pour la classification de document du 18ème siècle, ont développé des outils qui séparent le texte et l'image et servent également à la réduction du bruit, en se basant sur la transformée d'HERMITE et le filtre de GABOR. Leur technique est basée essentiellement sur l'identification des auteurs. Bien qu'elle a été testée sur des documents dégradée, l'approche commence par une réduction du bruit puis passe à une étape d'extraction des caractéristiques globales et d'analyse des formes. Les étapes de traitement sont illustrées dans le schéma suivant:

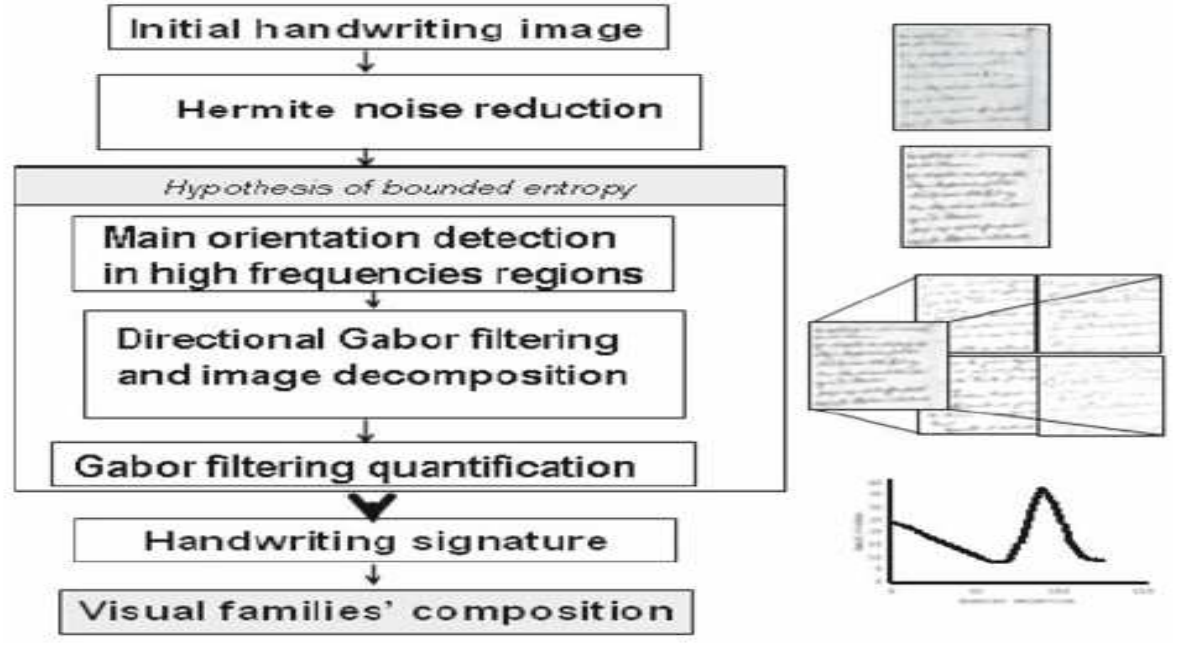


Figure I.9 Schéma global : réduction du bruit dans l'image [EGL07]

La réduction du bruit: la phase de réduction de bruit est primordiale à cause de la situation dégradée des documents anciens dû au flou surgis pendant la numérisation, les conditions d'archivage, le chevauchement entre les lignes, les taches d'encre...etc.

En général, les taches d'encre et le texte apparent du verso, ont un contraste faible par rapport

au texte original,

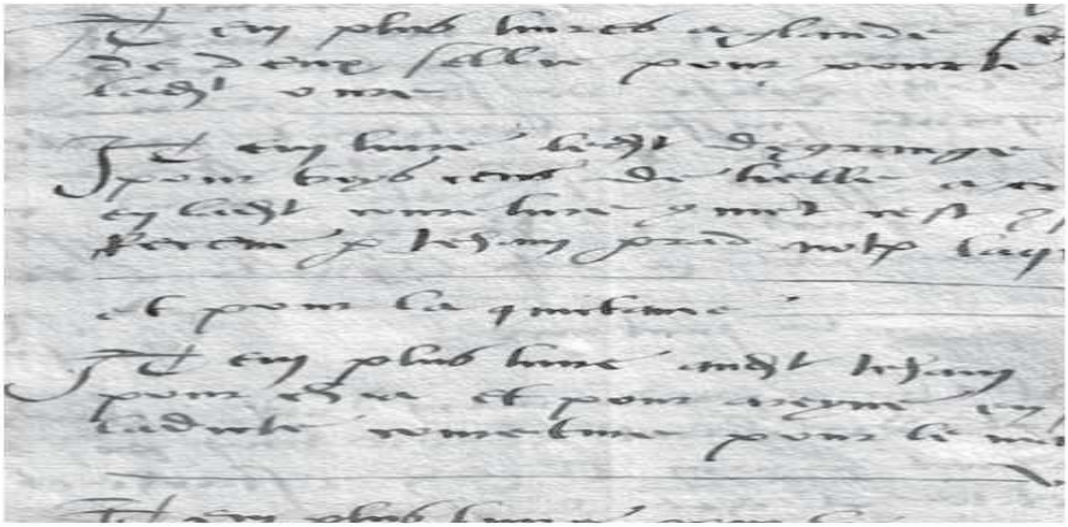


Figure I.10 Exemple d'un ancien document [EGL07]

En utilisant la transformée de HERMITE le système sépare le texte avec faible contraste de celui qui a un fort contraste, et en gardant ce dernier.

Le coefficient d'HERMITE est calculé comme suit :

$$C_{i,j}(x,y) = \begin{cases} \text{sign}\,(C_{i,j}(x,y)) \times (|C_{i,j}(x,y)| - \sigma_{i,j}(x,y) \times k_{ij}) \\ \\ 0 \quad \text{si} \ |C_{i,j}(x,y)| < \sigma_{i,j}(x,y) \end{cases} \qquad (58)$$

$$\text{et} \qquad \sigma_{i,j}(x,y) = \sigma_{i,j} \times (1 - M_{ij}(x,y)) \qquad (59)$$

Où :

$C_{i,j}(x,y)$ est le coefficient à la position (x,y) au quadrant (x,y) voir figure

$\sigma_{i,j}$ est le niveau pondéré du bruit dans le quadrant (i,j)

(i, j) est la zone considérée pouvant contenir le texte

M_{ij} (x, y) est un masque normalisé à la position (x,y)

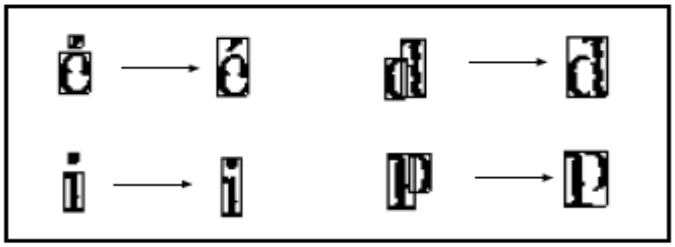
k_{ij} est un coefficient normalisé qui doit garder toujours la valeur maximale du quadrant (i,j)

2.2 Les approches analytiques

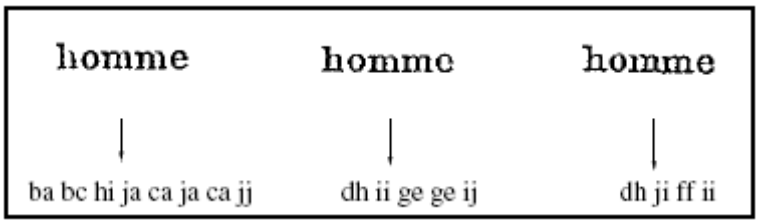
2.2.1 Indexation et recherche des mots dans les documents anciens :

Dans cette approche le mot est divisé en objets, chaque objet est identifié par un code et représente un caractère. Enfin chaque mot sera composé d'une combinaison de codes dont en

se base durant la phase de recherche. Pour ce faire le texte est segmenté en mots, puis l'extraction des caractères se fait en recherchant les composants connectés, il est bien à noter ici que ce travail est appliqué sur les écritures latines détachées, chaque mot ensuite sera étiqueté et enregistré en tant qu'indexe [MAR03]. Voir figure III.15



a-



b-

Figure III.15 a- Extraction des caractères b- Codage des mots

* Représentation normalisée des mots : d'après l'image on constate que le même mot peut avoir différents codes, cette contrainte est causée par: l'écriture détachée d'un caractère et l'attachement de deux caractères. Comme le montre la figue III.15, le mot « homme» peut avoir plusieurs codes.

L'étiquetage des caractères se fait par le biais d'un système neuronal supervisé, se qui fait que chaque mot sera doté d'un code unique. Pour se faire chaque mot est divisé en tranches, ensuite le système codifie le caractère inclus dedans. (Figure III.16)

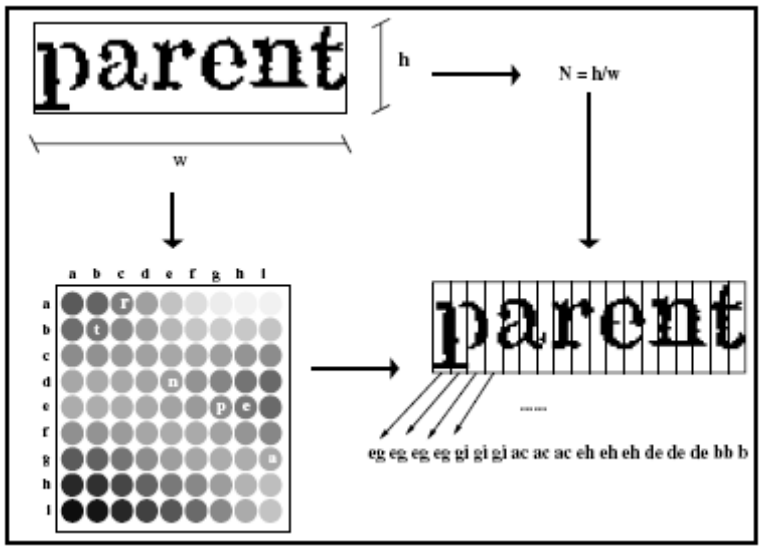


Figure III.16 Codage des mots

* Recherche des mots :

La recherche s'établit en deux phases, premièrement l'utilisateur envoie sa requête de recherche puis le mot recherché sera codifié et comparé avec les mots enregistrés dans la base des index, (figure III.17). La comparaison se fait entre les vecteurs caractéristiques, des mots calculés durant la phase d'indexation, en utilisant la distance Euclidienne.

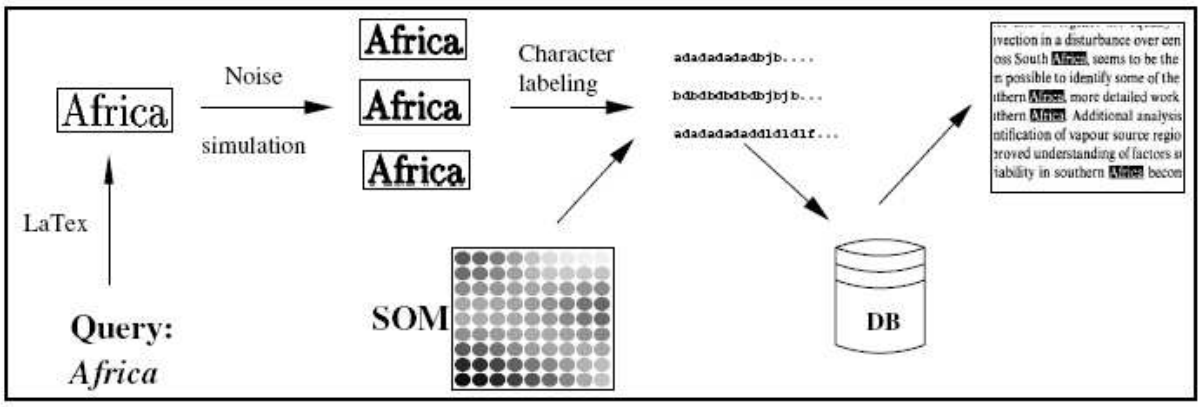


Figure III.17 processus de recherche

2.3 Les approches structurelles

2.3.1 Analyse des Orientations pour la Caractérisation d'Images de Documents de la Renaissance :

L'analyse des orientations pour la caractérisation des images est une approche proposée dans [JOU05] afin de caractériser des images de documents imprimés extraits d'un vaste corpus de la renaissance et actuellement numérisés au CESR de Tours figure III.18. Cette méthode est non seulement robuste aux bruits fréquemment rencontrés dans ce genre d'images (détériorations de l'encre, défauts de numérisation…) mais elle se veut aussi complètement indépendante de la typographie employée, de la taille des caractères, de la présence de parties graphiques…etc. C'est une contribution à l'analyse de la mise en page de documents par une séparation texte/graphique basée sur une localisation et un marquage en zones d'intérêt. L'originalité de ce travail vient de sa grande généricité (applicabilité à des documents de structures très hétérogènes), de l'absence quasi-totale de connaissances a priori sur les propriétés des textes, des graphiques et de leur agencement et de sa grande robustesse aux variations locales de contrastes, aux changements de résolution, aux phénomènes fréquents d'inclusion de texte dans les graphiques

Figure III.18 Exemples de pages de documents anciens actuellement archivés au CESR.

* Caractérisation des zones d'intérêt par extraction des orientations : Ce système d'indexation robuste et adapté à un corpus d'images au contenu hétérogène. Premièrement le système extrait visuellement la mise en page du document en séparant grossièrement les zones de texte (caractères d'imprimerie) des zones de dessin (lettrines, enluminures…). Les images sont redimensionnées à une taille fixe de (900x600). Le fond est ensuite identifié à l'aide d'un algorithme basé sur l'étude de l'homogénéité des niveaux de gris. Le marquage des pixels est réalisé à l'aide d'une fenêtre d'analyse de taille fixe que l'on déplace sur l'image. A chaque déplacement, les orientations caractéristiques de la zone sont extraites et analysées afin de labelliser chaque pixel inclus dans la fenêtre comme étant un pixel appartenant à du texte ou à un dessin. Cette technique de marquage implique que les pixels peuvent être marqués plusieurs fois.

L'extraction des orientations d'une zone de l'image se fait à l'aide d'une fonction d'auto-corrélation comme suit :

$$C_{xx}(k,l) = \sum_{k'=-\infty}^{\infty} \sum_{l'=-\infty}^{\infty} x(k',l') \cdot x(k'+k, l'+l) \qquad (60)$$

La rose des directions permet d'effectuer une analyse de cette fonction d'auto-corrélation. Cette rose donne, avec une grande précision, les orientations privilégiées présentes dans le bloc étudié. La rose est en fait un diagramme polaire. Soit (u,v) le point central de l'auto-corrélation et θi l'orientation étudiée, on calcule alors la droite Di tel l'ensemble de ses points (a,b) respecte la relation suivante: angle formé par les deux droites i $(a,b)(u, v) = \theta$. Pour chaque orientation θi on calcule ainsi la somme des différentes valeurs de la fonction d'auto-corrélation

$$R(\theta_i) = \sum_{D_i} C_{xx}(a,b) \qquad (61)$$

Ces valeurs sont ensuite normalisées pour ne garder qu'un aspect relatif de la contribution de chaque orientation.

$$R'(\theta_i) = \frac{R'(\theta_i) - R_{min}}{R_{max} - R_{min}} \qquad (62)$$

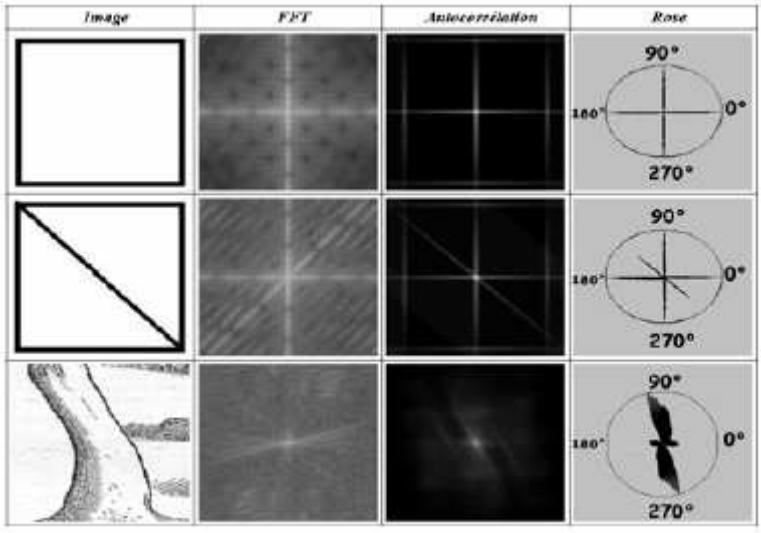


Figure III.19 Exemples de roses des directions

* Extraction de la mise en page : La rose possède des propriétés intéressantes. En effet, quand la zone étudiée possède une répartition homogène des directions, la rose produite est une

boule parfaite. Si on ajoute une ligne à l'image précédente (figure III.19), le fait que la rose soit normée génère deux caractéristiques invariantes: deux pics horizontaux de même intensité (pour 0° et 180°) et une même intensité pour toutes les autres directions.

Une autre propriété intéressante est que la rose est robuste au changement de résolution. Quelque soit la résolution à laquelle est calculée la rose, sa forme ne change pas (figure III.20).

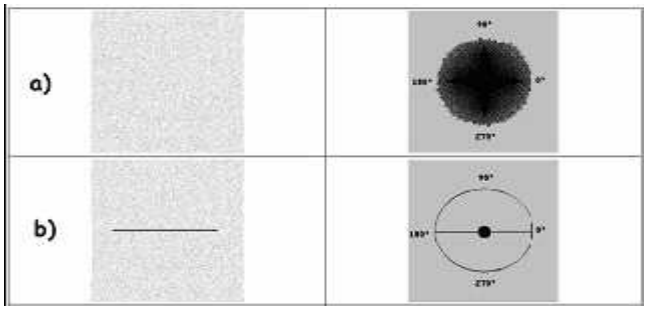


Figure III.20 Caractéristiques de la rose

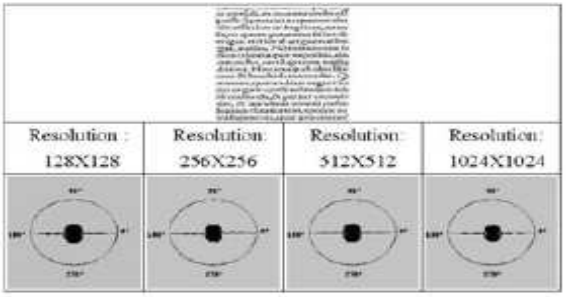


Figure III.21 Comportement de la rose à différentes résolutions

Si la zone de texte est homogène (espace interligne constant, taille des caractères constants…) elle possède les mêmes propriétés que l'image de la figure III.21

Lorsque la zone n'est pas homogène (différentes taille de texte, espaces interligne variable…), la taille de La « boule » peut diminuer voire même disparaître (figureIII.22).

Les deux autres formes de roses permettent d'identifier une zone de dessin. Ce type de rose est caractérisé soit par l'absence d'orientations horizontales, soit par la présence d'une direction horizontale mais sans la présence de pic.

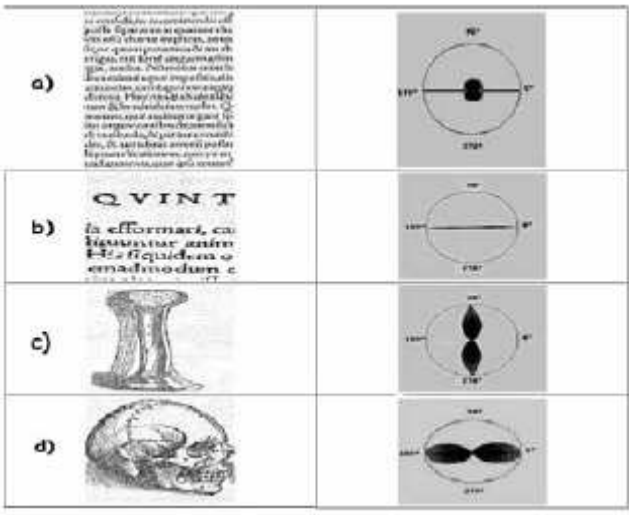


Figure III.22 Détermination des zones graphiques

2.3.2 Analyse structurelle des documents imprimés anciens:

Les documents anciens latins du dix-septième siècle sont différents par rapport aux documents actuels, Dans la plupart entre eux, on trouve la partie texte majoritaire par rapport au dessin, dont on trouve souvent une partie notée sur les marges. De l'autre part, les dessins se font avec des tailles différentes. Les textes sont structurés, titres, sous titres et paragraphes. En bas de page, on trouve le numéro de la page ou le premier mot de la page suivante. Quant au style, il est soit normal, justifié ou aligné à gauche. (Figure III.23)

Figure III.23 Exemple de pages de documents historiques

Les caractéristiques structurelles mentionnés dans [RAM07] sont:

1- Structure complexe des pages.

2- Le style est inidentifiable.

3- On trouve des notes manuscrites dans les marges.

4- Le numéro de page en bas.

5- Utilisation de plusieurs fontes.

6- Utilisation de lettrine, et des bordures spéciales.

7- Espace inconstant entre les caractères, les mots, et les paragraphes.

8- Les blocs de texte sont irréguliers.

9- Qualité dégradée des documents causant des images bruitées.

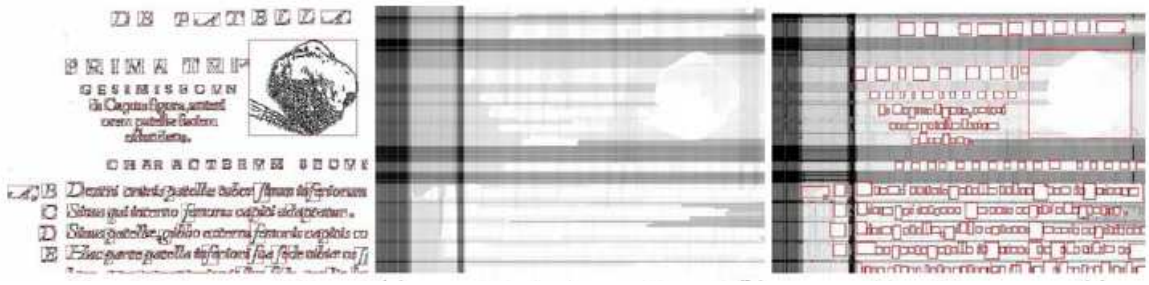
* Segmentation : la segmentation sert à isoler le texte du fond, la méthode utilisée dans ce contexte est une méthode hybride qui fait une analyse descendante, et une analyse ascendante [BEL97].

Représentation de l'avant plan : pour l'extraction de l'avant plan (texte ou graphique) il faut rechercher les composants connectés, l'avant plan peut être texte, graphique ou bruit, la taille du cadre englobant l'objet permet d'identifier son contenu. Donc après avoir effectué une binarisation tout le contenu sera cadré dans un rectangle, ce dernier sera enregistré dans une liste. Les formes déduites sont les suivantes:

1- Bruit : si le cadre a une très petite taille

2- Le texte : si le cadre a une petite taille

3- L'image : si le cadre est d'une grande taille (figure III.24)



(a) (b) (c)

Figure III.24 a- détection avant plan, b- détection arrière plan c- superposition de a et b

* Représentation de l'arrière plan : le fond de chaque document est représenté souvent par les pixels blancs, un nombre de pixels aligné horizontalement ou verticalement sépare deux mots, deux, paragraphes ou parfois même deux caractères. Par conséquent pour chaque pixel p(i,j) les fonctions suivantes sont définies [RAM07] :

lgb_h(i, j) la fonction qui calcule le nombre de pixels successifs horizontaux

lgb_v(i, j) la fonction qui calcule le nombre de pixels successifs verticaux

Max=hauteur de l'image x largeur de l'image

$$Ng(i, j) = Max - [lgb\_v(i, j) + lgb\_h(i, j)] \quad (63)$$

* Séparation texte/graphique : la séparation texte / graphique se fait à l'aide de l'algorithme suivant :

1- Une zone texte à l'intérieur d'une zone graphique sera étiquetée *Text_Graphic*
2- Le texte et les blocs étiquetés sont colorés en blanc durant la phase de binarisation
3- Représentation de l'arrière plan

Extraction de zones de texte : en utilisant les informations extraites précédemment sur l'arrière et l'avant plan on peut effectuer une séparation texte/graphique, au début, les composants connectés et labellisés « Texte » sont réunis afin de reconstruire les paragraphes. Pour regrouper deux zones « Texte », il faut qu'ils soient d'abord fermés, et que la distance entre leurs centres de gravités vérifie la condition suivante :

$$D(G_1,G_2)x(256-\min[Ng(i,j)]),i,j\in(G_1,G_2) \leq \text{seuil de fusion} \quad (64)$$

Où d est la distance Euclidienne entre les deux centres de gravité.

3. conclusion :

Les documents anciens ont ouvert un sous domaine de recherche avec beaucoup de défis aux chercheurs. Les méthodes « classiques » de traitement, d'indexation et de modélisation des images ne sont pas directement applicables sur les documents historiques.

A travers ce chapitre nous avons essayé de mettre le point sur les tendances actuelles de la recherche.

1- Introduction

Le traitement d'image nous aide à segmenter et à reconnaître les divers objets qui la composent à partir de leur seule silhouette. Ce traitement est utilisé particulièrement dans les applications de tri dans lesquelles on est intéressé à saisir ou trier des objets arrivant sur un convoyeur, dans les applications de reconnaissance de caractères, ainsi que pour des applications de surveillance ou de guidage.

L'utilisateur doit disposer d'une représentation de la forme vérifiant plusieurs propriétés

1. une bonne fidélité à la forme initiale,
2. une bonne discrimination de formes différentes,
3. une bonne adaptation aux opérations de reconnaissance des formes

De nombreuses représentations des formes ont été développées concurremment, chacune pour répondre à un problème parfois assez spécifique et donc mettant l'accent sur l'une ou l'autre des propriétés ci-dessus.

L'une des méthodes de représentation de forme est celle qui la représente par son contour. Plusieurs techniques on été proposées dans littérature et qui ont été largement utilisées dans la reconnaissance des formes. Tels que les moments de Fourrier, les modèles de Markov, l'approximation Spline, l'approximation polygonale, la chaîne de code…etc.

Dans ce chapitre nous allons expliquer brièvement ces méthodes en essayant de montrer leurs apports dans le domaine de reconnaissance de forme.

2- approximation polynomiale : on distingue trois type d'approximation polynomiale :

Les approximations analytiques par ajustements de nuages de points par des polynômes du premier ordre, les approximations polygonales par des critères géométriques obtenus en parcourant la courbe des points ordonnés selon une abscisse curviligne, et les approximations par des polynômes de degré > 1, et en particulier les approximations par des fonctions splines.

* Approximation d'un nuage de points par une droite unique : soit $M_i = (x_i, y_i)$ un ensemble de points en nombre N, ce qu'il faut faire ici c'est d'approcher ces points par une droite, pour ce faire, nous avons deux possibilités :

2.1 Approximation par régression linéaire : C'est une approche aux moindres carrés : on recherche la droite

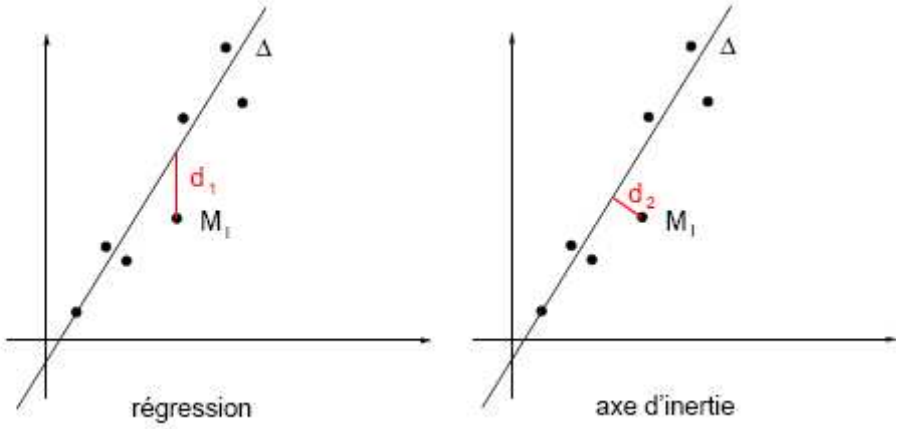
Δ : $y = a_0 + a_1 x$ qui minimise la distance :

$$D^2_1 = \sum_i^n [y_i - (a_0 + a_1 x_i)]^2 \qquad (65)$$

La solution est donnée par : $A = x^{\#} y = (x^t x)^{-1} x^t y$ où: $x^{\#}$ dénote la matrice pseudo-inverse de la matrice x où x, y et A sont donnés par :

$$X = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ .. & .. \\ 1 & x_n \end{pmatrix}, \qquad Y = [y_1, y_2, \ldots, y_n]^t, \qquad A = [a_0, a_1]^t$$

Ces formules s'étendent très aisément aux espaces de dimensions supérieures, ainsi qu'aux approximations par des polynômes d'ordre plus élevé. La distance minimisée est celle mesurée selon le seul axe y. C'est donc une mesure généralement mal adaptée en traitement d'image, puisque x et y jouent habituellement un rôle équivalent. On préfère donc les méthodes par axe d'inertie.



FigureIV.1 Deux approximations linéaires aux moindres carrés minimisant des distances différentes : à gauche par régression linéaire, à droite par axe d'inertie.

2.2 Approximation par axe principal d'inertie : C'est aussi une approche aux moindres carrés, mais on minimise dans ce cas la somme des distances de tous les points à la droite Δ :

$$d^2_2 = \sum_{i=1}^{n} \frac{(a_0 + a_1 x_i - y_i)^2}{a^2_1 + 1} \qquad (66)$$

C'est l'équation de l'axe d'inertie des points, qui passe par leur centre de gravité x_g , y_g et qui est donnée comme vecteur propre de plus grande valeur propre de la matrice de forme quadratique :

$$S = \sum_{n} V_i V_i^t = \begin{pmatrix} \sum x_i x_i & \sum x_i y_i \\ \sum x_i y_i & \sum y_i y_i \end{pmatrix} \qquad (67)$$

Ces équations s'écrivent sans problème en dimensions supérieures (pour estimer des variétés d'ordre variable).

Elles se transcrivent beaucoup plus difficilement à des polynômes d'ordre plus grand car on ne sait pas, en règle générale, exprimer la distance d'un point courant à une telle fonction [MAI03].

3. Approximation polygonale :

La description polygonale d'une forme consiste à réduire une courbe continue ou finement échantillonnée à une ligne polygonale. Pour ce faire, plusieurs algorithmes ont été proposés dans la littérature dont voici les plus connus :

3.1 Algorithme de la corde ou de Ramer

C'est un processus de subdivision qui peut être entrepris soit de façon récursive (chaque segment créé fait l'objet d'une nouvelle subdivision), soit de façon itérative (la courbe est considérée globalement à chaque étape).

C'est cette dernière version que nous examinons sur l'exemple de la figure IV.2. Les sommets du polygone sont choisis successivement comme les points de ϒ les plus éloignés des cordes précédemment tirées. Le processus s'arrête lorsque la nouvelle distance candidate est inférieure à un seuil fixé. Très employé dans de nombreuses applications par la simplicité de sa mise en oeuvre, l'algorithme de la corde n'est pas très rapide. Il ne garantit pas non plus une convergence uniforme vers la courbe finale car, dans son implémentation itérative, il se peut que la distance à l'étape n, soit supérieure à celle à l'étape n-1.

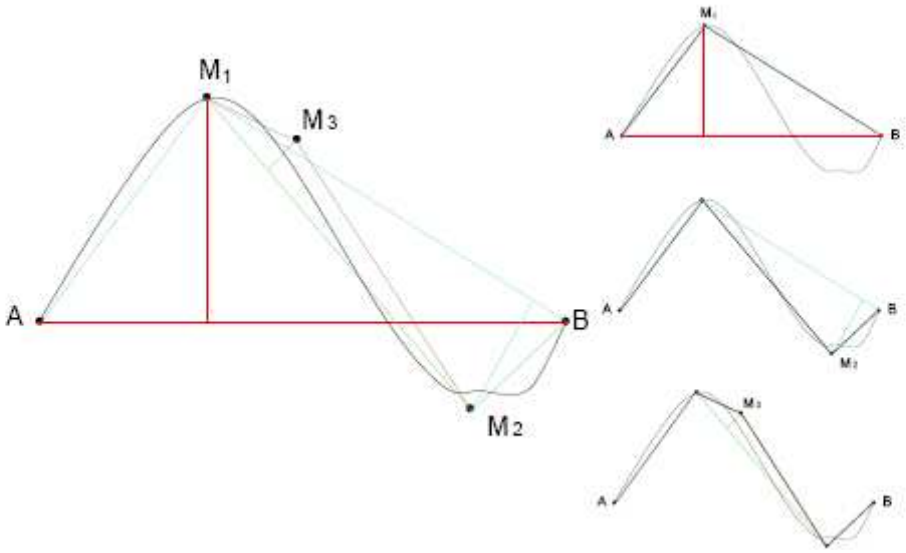


Figure IV.2 Algorithme de la corde. Les points M1, M2 et M3 _ sont successivement sélectionnés pour créer la ligne polygonale représentant la courbe AB Le critère de sélection est la distance maximale à la corde polygonale précédemment obtenue.

3.2 Algorithme de Dunham

Dans cette méthode on cherche à supprimer les cordes successives qui sont presque alignées. On se fixe donc une tolérance angulaire € Partant de A, on couvrira par une même corde tous les points de la courbe qui se trouvent dans la tolérance et l'on choisira le premier point du polygone comme le dernier point de cet ensemble. Le processus est répété en ce point.

3.3 Algorithme de Wahl et Danielsson

Son objectif est de minimiser l'aire laissée entre la courbe et la ligne polygonale.

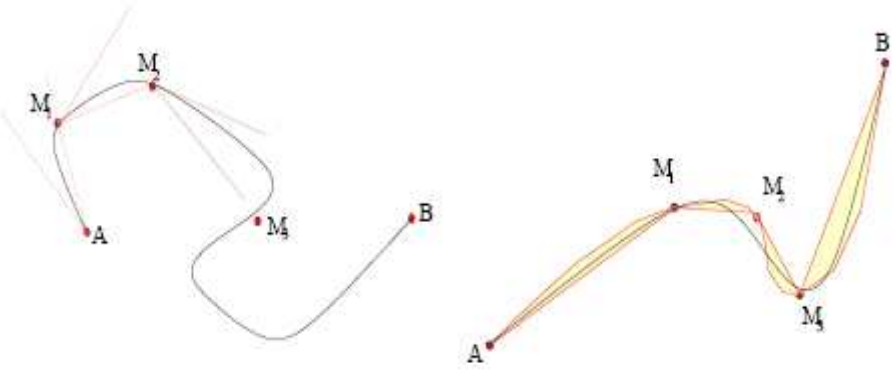


Figure IV.3 Algorithme de Dunham, à gauche, les points M1, M2 et M3 sont choisis lorsque la tolérance angulaire € est insuffisante à approximer la courbe. A droite, algorithme de Wahl et Danielsson : il cherche un minimum des surfaces laissées entre la ligne polygonale et la courbe.

3.4 Algorithme progressif

Enfin, un algorithme progressif utilise le critère de distance de tout point de la courbe au segment d'approximation, mais dans un schéma progressif. Dans ce schéma, partant d'une extrémité, la courbe est parcourue, et le point courant est retenu comme extrémité d'un segment d'approximation s'il est le dernier point visité permettant une approximation de tous les points déjà visités, à une tolérance € donnée.

4. Approximation par des Splines : l'idée des Splines est d'approximer le contour par des graphes d'ordre supérieur à une droite (degré ≥1) le choix du degré du polynôme se fait selon les propriétés de continuité au contour obtenu :

Les segments de droite (degré 1) assurent la continuité du contour, les approximations par polynômes d'ordre 2 permettent d'avoir des dérivées continues, et les polynômes d'ordre 3 permettent d'avoir des courbures continues. Deux types de fonctions peuvent être utilisées:

1. les fonctions interpolantes qui passent exactement par tous les points de contours qui sont utilisés pour calculer la Spline ;

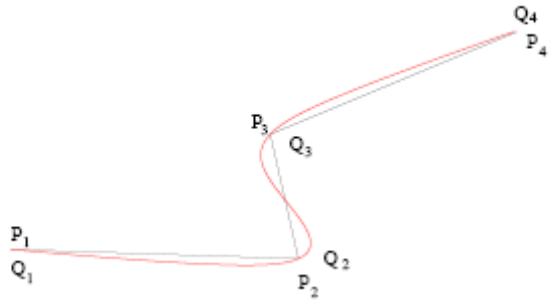
2. les fonctions approximantes qui ne passent pas nécessairement par les points mais qui s'en approchent de façon contrôlée.

* Approximation : Dans le cas des fonctions approximantes, on définit la Spline par morceaux en fonction d'une variable continue μ, et des m points de contrôle Pi qui la déterminent. Qi et Pi sont des fonctions de même dimension :

Des vecteurs de $\square^2$ si les points Pi sont définis par leurs coordonnées du plan ou des vecteurs de $\square^3$ dans l'espace. Le morceau i est défini par :

k-1

$$Q_i(u) = \sum_{r=0} P_{i+r}\, b^k_r(u) \qquad i = 0,\ldots, m-k+1 \qquad (68)$$



FigureIV.4 Dans le cas d'une interpolation Spline, les jonctions entre les morceaux de Splines sont les points de contrôle eux-mêmes.

***** Interpolation

Dans ce cas, les coefficients des Splines de l'équation sont inconnus et l'équation devient :

$$Q_i(u) = \sum_{r=0}^{k-1} C_{i+r}\, b^k_r(u) \qquad i = 0,\ldots, m-k+1 \qquad (69)$$

Et les coefficients inconnus Ci sont déterminés en contraignant les Splines à passer par les points de contrôle Pi

$$P_i(0) = \sum_{r=0}^{k-1} C_{i+r}\, b^k_r(0) \qquad (70)$$

5. La transformée de fourrier : Le contour d'une forme généré par le code de Freeman sera caractérisé par :

- les valeurs : entre 0 et 7
- la direction : de 0 à 2π par pas de $\pi/4$ en suivant le sens contraire de l'aiguille d'une montre.
- La longueur : 1 ou $\sqrt{2}$ suivant la parité des codes.

La transformée de Fourier peut être calculée sur une fonction temporelle périodique générée à partir de ces caractéristiques. La notion du temps est associée à la durée du parcours d'une

portion du contour à partir d'un certain point de départ [SNO00]. Par exemple, le temps nécessaire pour traverser les *k* premiers arcs qui relient les points du contour est :

$$t_k = \sum_{i=1}^{k} \Delta t_i \qquad (71)$$

Avec : $\Delta t_i = 1$ si le code est pair, $\sqrt{2}$ s'il est impair

Si le contour est décrit par N codes, il sera périodique de période :

$$T = \sum_{i=1}^{N} \Delta t_i \qquad (72)$$

A partir de cette description le contour peut être décrit par une fonction continue (x(t) , y(t)). les distances parcourues à l'instant tk sont décrites par x(k) (projection du contour sur l'axe des X) et y(k) (projection du contour sur l'axe des Y) avec :

$$x_k = \sum_{i=1}^{k} \Delta x_i \quad (73) \quad \text{et} \quad y_k = \sum_{i=1}^{k} \Delta y_i \quad (74)$$

Δx_i et Δy_i peuvent être 1, 0 ou –1 suivant la valeur et la direction du code de Freeman qui relie deux points du contour. La figure IV.5 présente ce codage.

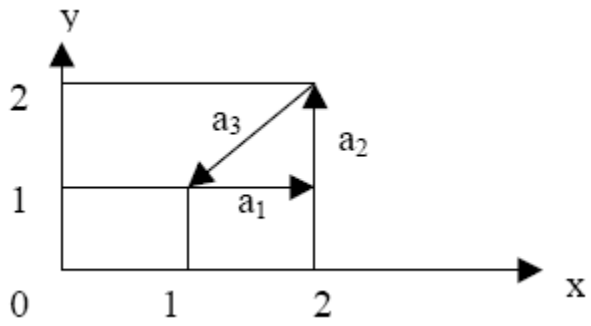


Figure IV.5 : projection des codes de Freeman sur l'axe des X et l'axe des Y
$a_{1x}=1$, $a_{1y}=0$; $a_{2x}=0$, $a_{2y}=1$; $a_{3x}=-1$, $a_{3y}=-1$.

Ainsi, le contour peut être décrit par la fonction discrète (x(k), y(k)). La Transformation de Fourier au rang N peut être écrite de la manière suivante :

$$X_N(k) = A_0 + \sum_{n=1}^{N} a_n \cos \frac{2n\pi k}{N} + b_n \sin \frac{2n\pi k}{N} \qquad (75)$$

$$Y_N(k) = C_0 + \sum_{n=1}^{N} c_n \cos \frac{2n\pi k}{N} + d_n \sin \frac{2n\pi k}{N} \qquad (76)$$

Avec : k, le $k_{ème}$ point du contour, N : est le nombre des harmoniques nécessaires dans l'approximation du contour par les coefficients de Fourier. Il est facile de montrer que lorsque N est égal au nombre de points du contour, l'approximation du contour décrit de façon complète le contour initial. Par suite, une reconstitution fidèle à partir des coefficients du Fourier est immédiate par la transformée de Fourier inverse [OPP95].

a_n, b_n, c_n et d_n sont les coefficients de Fourier correspondant au $n_{ième}$ harmonique.

A_0 et C_0 sont dits les composantes continues correspondant aux points initiaux là où la fréquence est égale à 0. Ces coefficients sont générés par l'algorithme de Transformée de Fourier rapide connu sous le nom (FFT : Fast Fouried Transform). L'idée de la Transformée de Fourier rapide consiste à décomposer la Transformée de Fourier Discrète en un sous-ensemble de Transformée de Fourier Discrète dont le calcul des coefficients de Fourier discret est moins coûteux. Cette décomposition est assurée en considérant le nombre N d'échantillons du signal comme une puissance de 2 : $N = 2^r$. Etant donné que N est pair, on peut calculer X(k) en séparant x(n) en deux sous-ensembles de N/2 points chacun. Le premier sous-ensemble contient les N/2 points pairs, le deuxième contient les N/2 points impairs.

A partir de deux équations précédentes on peut écrire $(X_n(k), Y_n(k))$ de la manière suivante :

$$X_n(k) = a_n \cos\frac{2n\pi k}{N} + b_n \sin\frac{2n\pi k}{N} \qquad (77)$$

$$Y_n(k) = c_n \cos\frac{2n\pi k}{N} + d_n \sin\frac{2n\pi k}{N} \qquad (78)$$

Ce qui implique que :

$$\frac{X_n^2(k)}{a_n^2 + c_n^2} + \frac{Y_n^2(k)}{b_n^2 + d_n^2} = 1 \qquad (79)$$

Donc tous les points $(X_n(k) , Y_n(k))$ appartiennent à une forme elliptique

L'approximation du contour par les coefficients de Fourier peut être vue comme une superposition de formes elliptiques de différentes tailles, positions et orientations. La figure IV.6 présente une illustration graphique de la $1^{ère}$, $2^{ème}$, $3^{ème}$ et $4^{ème}$ approximations harmoniques de Fourrier du contour du caractère « ba » ب

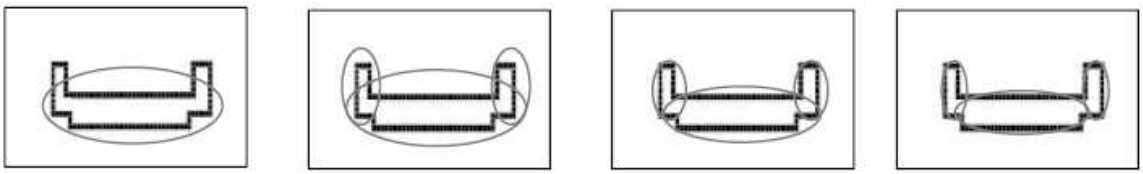


Figure IV.6. La superposition des coefficients de Fourier du caractère «ba ب»

6. Détection point contour : Dans cette catégorie, le contour de la forme est représenté par un ensemble de points (Figure IV.7). A chaque point est associé un descripteur qui le distingue par rapport aux autres. La comparaison entre deux formes sera donc basée sur les descripteurs, son principe est de trouver une correspondance entre les points contours de l'image source et l'image cible [ARI03].

Figure IV.7 Représentation du contour par un ensemble de points

Une étude récente propose un descripteur de contexte de forme pour calculer la similarité entre deux formes. Ce descripteur sera utilisé pour la reconnaissance des objets. Le contexte de forme à un point contour référence capture la distribution des autres points contours relatifs. Soit une forme avec N points, pour le point P_i, l'histogramme hi pour les N-1 points relatives est calculé par : $h_i(k)= \#\{q \neq p_i : (q-p_i) \in bin(k)\}$

Où bin(k) sont les cadres uniformes dans l'espace log-polaire (figure IV.8)

L'histogramme hi est le contexte de forme pour le point pi. Pour calculer la correspondance entre les points de deux formes on utilise un graphe de comparaison

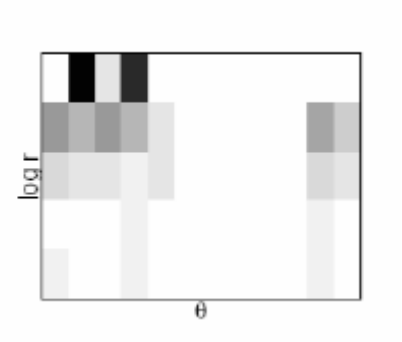


Figure IV.8 Contexte de forme pour les points contours

7. Suivi de contour par les HMM : pour faire une détection précise des points de contour des objets, plusieurs paramètres peuvent être appliqués tels que l'intensité, la couleur...etc. ces derniers seront intégrés dans des HMM [RAB86], les états des HMM présentent les points contours réels de chaque ligne notés par : $S=\{S_1,S_2,\ldots S_\phi,\ldots S_m\}$, les observations des HMM $O=\{O_1,O_2,\ldots O_\phi,\ldots O_m\}$ sont collectés pour chaque ligne ϕ. Le modèle de MARKOV sera spécifié par :

1- le nombre des états
2- le modèle d'observation $p(O_\phi / S_\phi)$
3- les transitions de probabilité $p(S_\phi / S_{\phi-1})$
4- l'état S_ϕ et l'observation O_ϕ sont indépendants, respectivement du précédent état $S_{\phi-1}$ et précédente observation $O_{\phi-1}$
5- La propriété Markovienne vérifie que $p(S_\phi / S_1, S_{2,\ldots} S_{\phi-1}) = p(S_\phi/S_{\phi-1})$

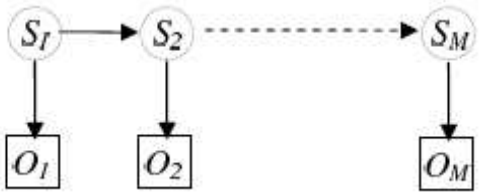


Figure IV.9 : modèle graphique de suivi de contour

L'observation dans la ligne ϕ noté par O_ϕ peut inclure plusieurs paramètres par exemple l'intensité des pixels noté $\rho_\phi(\lambda)$ et la détection du contour Z_ϕ.

Pour la détection du contour soit J, qui représente le nombre de contours trouvés

($Z_{\phi=}$ Z_1,$Z_{2,...}$ Z_j). Un parmi les j contours trouvés représente le contour réel, pour cela nous pouvons supposer j+1 hypothèse :

H_0 ={e_j=F, j=1..J}

H_j={ e_j=T, e_k=F, k=1..J, k ≠ j}

e_j=T signifie que le contour j est un contour réel, par contre e_j=F signifie que le contour j n'est pas un contour réel.

L'hypothèse H_0 signifie qu'aucun contour réel n'a été trouvé

Le modèle de probabilité contour est donné par :

D'autres paramètres tels que la couleur du premier plan et de l'arrière plan des régions peuvent facilement être intégré avec les HMM. Soit p(v / FG) et p(v / BG) qui représentent respectivement la distribution des couleurs dans le premier plan, et la distribution des couleurs dans l'arrière plan. Les probabilités postérieurs p(FG / v) et p(BG / v) peuvent être donné par :

$$P(BG\backslash v) = \frac{P(v\backslash BG)}{P(v\backslash BG) + P(v\backslash FG)} \qquad (80)$$

$$P(FG\backslash v) = \frac{P(v\backslash FG)}{P(v\backslash BG) + P(v\backslash FG)} \qquad (81)$$

8. Chaîne de code : la chaîne de code a été proposée initialement par Freeman en 1961 [FRE86], connue par sa simplicité et son efficacité, ce code est constitué par une liaison d'un ensemble de pixels dont le dernier sera marqué par un mouvement inversé Figure IV.10

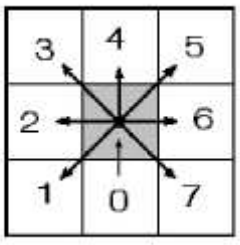


Figure IV.10 : Mouvement élémentaire de Freeman pour la chaîne de code

Le codage 8-conneté est souvent le plus préféré et nécessite 3 bits par lien, par contre le codage 4-connecté nécessite 2 bits par lien mais démontre ses limites devant des formes assez complexes.

La chaîne de code a été largement utilisée dans plusieurs applications, notamment : la description géométrique des formes, le codage des régions, la reconnaissance des manuscrits, la reconnaissance des formes, la compression des documents binaires …etc.

* Propriétés des chaînes de Freeman:

Les chaînes de Freeman se prêtent à un certain nombre de manipulations commodes.

1. On obtient une dilatation de la courbe d'un facteur k en répétant k fois chaque descripteur.

2. On ne peut généralement pas réduire une courbe sans distorsion.

3. On fait tourner une courbe de k x ($2\pi/n$) (dans le cas d'une chaîne de Freeman en n-connexité) en ajoutant (ou retranchant) k modulo n à la chaîne initiale.

4. On mesure la longueur d'une courbe par les formules suivantes :

– en 4-connexité : L= nombre de descripteurs,

– en 8-connexité : L= nombre de descripteurs pairs + $\sqrt{2}$ x nombre de descripteurs impairs.

5. Inversion d'un chemin : on inverse tous les descripteurs et on inverse la séquence. L'inverse d'un descripteur j est $j' = n/2 + j \bmod (n)$.

Exemple en 4-connexité : j= {001321}, j'= {301322}

6. Simplification d'un chemin : c'est un chemin dont on a supprimé des détails sans changer globalement la forme. Cela s'obtient en remplaçant des séquences de p descripteurs consécutifs par des descripteurs équivalents reliant les mêmes points.

Exemple : en 4-connexité {012} devient {1}, en 8-connexité {03} devient {2}

7. Réduction d'un chemin : c'est l'un des chemins de longueur minimale reliant les 2 extrémités de la courbe initiale. On associe 2 par 2 des descripteurs inverses de la chaîne et on les supprime.

Exemple en 4-connexité : x={00132122} devient {21}.

On obtient tous les chemins réduits en changeant l'ordre des associations.

En 8-connexité la réduction est un peu plus complexe car il faut aussi regrouper des ensembles de 3 ou 4 descripteurs qui s'annulent : exemple {025} ou {7225}

8. Fermeture d'un contour : on teste la fermeture d'un contour en vérifiant que la chaîne réduite est nulle. On ferme un contour en lui ajoutant le chemin inverse d'un de ses chemins réduits. Il y a beaucoup d'autres fermetures possibles que par addition de l'inverse d'un

réduit, mais les fermetures obtenues ainsi sont de longueur minimale. Il y a d'autres fermetures minimales que celles obtenues par addition de l'inverse d'un réduit.

* Codage de contours : le codage des contours est basé généralement sur deux aspects : la recherche des points de départ (têtes), et les mouvements (Figure IV.11). Dans son travail [RED07] suit la même démarche mais avec une représentation multi-échelles des contours. Dans notre cas on utilise une représentation à l'échelle normale.

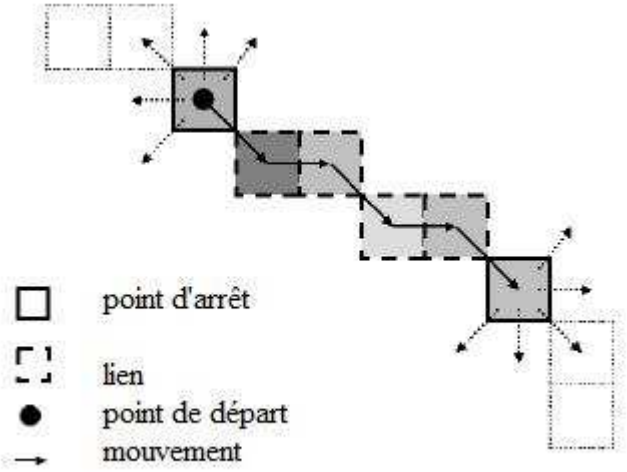


Figure IV.11 : Schéma de représentation des contours

Le codage des contours commence par une détection du point de départ qui est souvent le point le plus haut à gauche ou à droite dans l'image, ensuite une série de mouvements, désignant des passages d'un pixel à son voisin (direction), sera enregistrée (Figure IV.12).

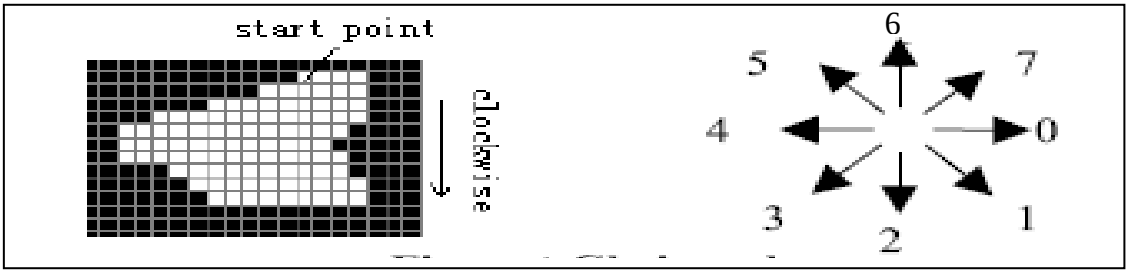


Figure IV.12 : choix point de départ et suivi du contour

Pour comparer deux formes il suffit de comparer leurs codes de Freeman correspondants, dans ce cas on ne cherche pas une similarité exacte mais seulement approchée, car, nous devons tenir en compte la présence de bruits lors de la détection des contours. Donc, on est amené à utiliser des techniques de distance d'édition pour lesquels on chiffre les coûts des

erreurs possibles: omission de contours, adjonction de contours, permutation de descripteurs, etc. On utilise alors généralement des techniques de programmation dynamique.

9. Conclusion : Dans ce chapitre, nous avons vu quelques approches qui traitent les contours des objets. En effet, beaucoup de ces représentations tirent profit de spécificités attachées à la famille particulière des objets à traiter pour une application donnée dans un contexte précis. On remarque, également, que toutes les approches essaient de montrer une simplicité de description de la forme. Particulièrement, La chaîne de code est la méthode la plus sollicitée grâce à sa simplicité et efficacité [RED07] et sa vaste implication dans divers domaines de reconnaissance de formes.

1. Introduction

La recherche d'information (RI) est un axe de recherche en pleine expansion à voir par le nombre de productions scientifiques, de consortiums, de projets nationaux et internationaux et de congrès dédiés à ce thème. La recherche d'images fait partie de la RI, mais possède ses propres spécificités. Bien évidement elle a bénéficié des évolutions extraordinaires du domaine néanmoins certaines technologies ont été développées spécialement pour les images.

Les anciens documents sur papier, qui sont généralement de mauvaise qualité, subissent une grande détérioration puisqu'ils ne sont pas bien conservés et ne sont pas protégés par une politique de restauration et de sauvegarde rigoureuses. Afin de remédier à la disparition de ce patrimoine culturel et scientifique de très grande importance, des compagnes de numérisation sont lancées partout dans le monde et des conventions internationales sont signées ici et là. On se retrouve donc avec de gigantesques bases de documents anciens scannés qu'on doit analyser, traiter, classer,… etc.

C'est dans ce cadre que rentre notre travail de recherche. L'objectif étant de réaliser un système permettant la restauration, l'indexation et la recherche des images de documents anciens. Notre choix s'est porté sur une méthode semi-automatique d'indexation et de recherche.

Dans notre approche, dans sa première version, le choix des indexes se fait manuellement tandis que leurs paramétrage et stockage se font automatiquement. L'idée de base est la recherche des indexes, détection de leurs contours, paramétrage des contours et stockage dans une base de données afin d'effectuer des recherches ultérieures. Le paramétrage des contours est une représentation sous forme de « séquence de codes, chaîne de code »**.** Cette représentation sera affinée par la suite pour résoudre le problème de la taille.

La phase de recherche se fait par l'algorithme de programmation dynamique DTW. Les résultats obtenus sont très encourageants.

2**. Description de l'approche**

Notre approche consiste à détecter les contours des indexes, les coder, optimiser leurs codes puis les ranger dans une base de données. Vu que notre objectif vise les documents anciens nous devons tenir en compte de leurs états, (variation du contraste, tâches d'encre…etc.). Notre système effectue d'abord: 1- une transformation de l'image couleur vers le niveau de gris, cette opération est suivie directement par une binarisation 2- on fait une détection du contour de l'image binaire résultante, 3- le contour sera codé en séquence de code, 4- la

séquence de code est affinée, optimisée, et rangée dans la base d'index. Le schéma présenté dans la figure V.1, illustre les étapes du processus d'indexation

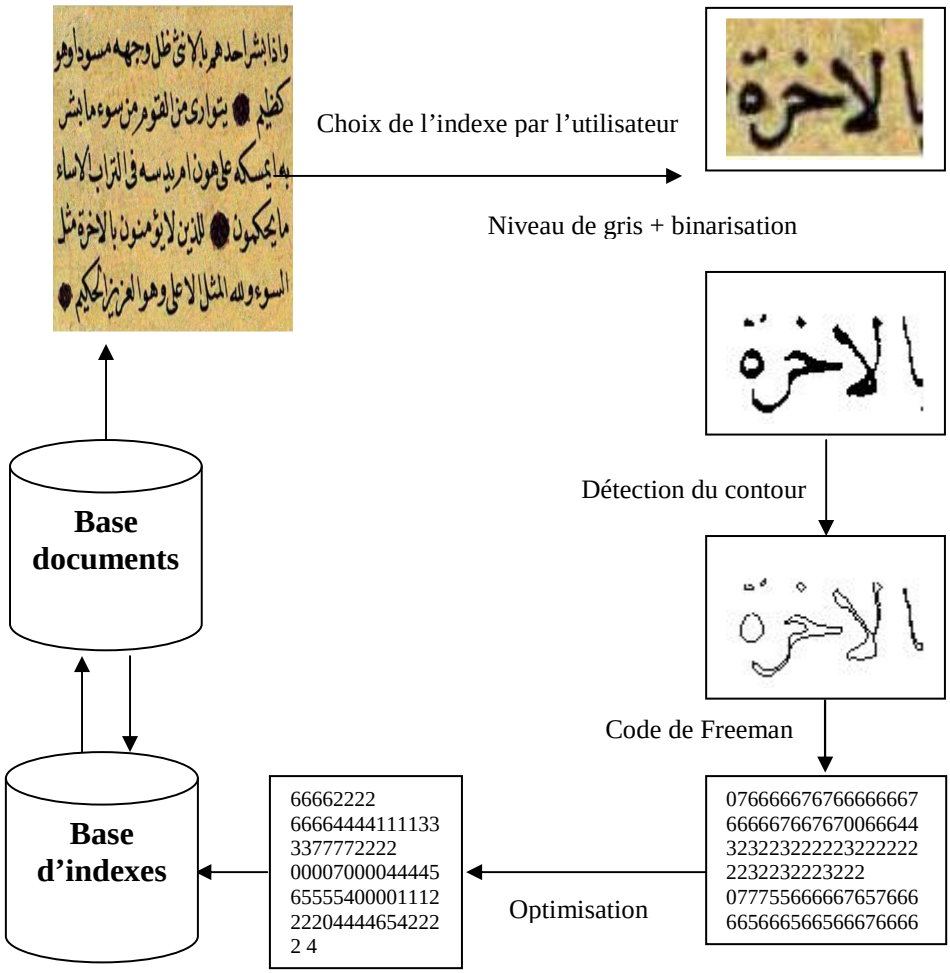


Figure V.1 Schéma du Processus d'indexation

2.1 Choix de l'indexe: consiste à sélectionner les mots appropriés, qui représentent le document, par des spécialistes.

2.2 Binarisation: La binarisation permet de passer d'une image en niveaux de gris à une image binaire. En général, on utilise un seuil de binarisation approprié qui traduit la limite des contrastes fort et faible dans l'image. Mais pour des images peu contrastées ou à contraste variable (i.e. la distribution de niveaux de gris n'est pas clairement bimodale), il est difficile de fixer ce seuil à une valeur précise. Dans notre approche nous traitons des documents archivistiques, nous sommes, généralement, donc face à des images de faible contraste, qui

peuvent contenir aussi des taches d'encre éparpillés sur les pages. La figure V.2 en donne un exemple.

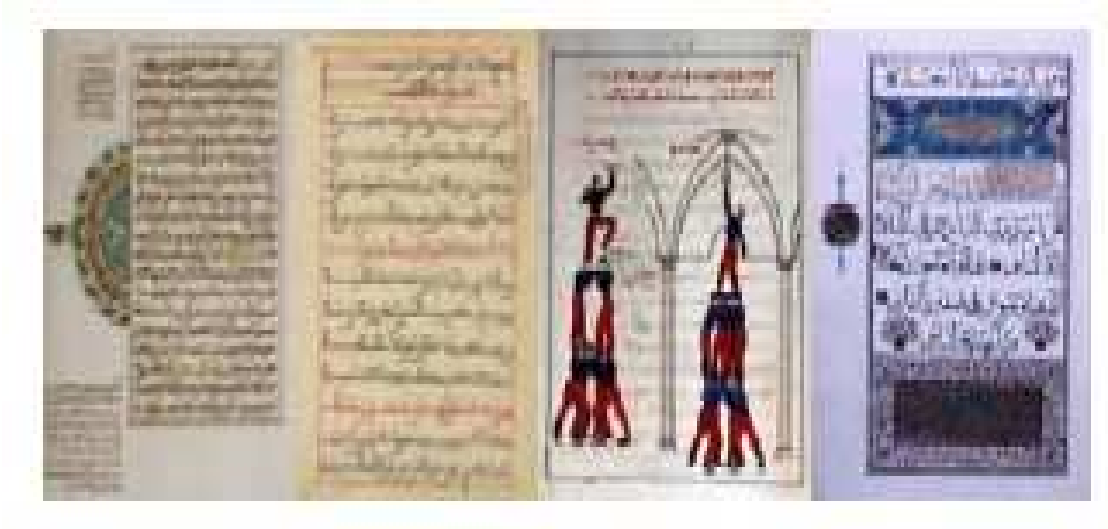


Figure V.2 Echantillon d'images historiques de faible contraste

Avant de procéder à la binarisation, nous avons commencé par la transformation des images couleurs en niveau de gris, pour ce faire, nous créons une matrice d'image I de taille LxM où : L est la longueur de la matrice et M la largeur de la matrice et I[L ,M] représente l'intensité lumineuse du pixel [L,M].

A chaque point couleur nous prenons les intensités RVB (rouge, vert et bleu) et nous calculons le niveau de gris selon la formule suivante : Gris=0.30 R+059 V+0.11 B.

Figure V.3 Transformation Niveau de gris + Binarisation d'une image

Plusieurs approches ont été proposées dans la littérature permettant de séparer le texte du fond, parmi elles :

La méthode d'OTSU [OTS76]: basée sur l'histogramme de l'image, utilise une analyse discriminante pour séparer le texte du fond, son principe consiste à diviser l'image en classe puis chercher le seuil optimal en minimisant la fonction variance inter et intra classes.

La méthode de KITTLER [KIT86] : cette méthode commence par une sélection d'un seuil aléatoire T, puis calcule l'erreur minimale de T qui sera utilisé comme seuil optimal.

La méthode de NIBLACK [NIB86]: basée sur un seuillage local, à chaque point pixel (x,y) le seuil est calculé en utilisant l'écart type dans un voisinage local.

Ces trois méthodes sont les plus efficaces destinés vers la séparation texte/fond

Dans notre travail nous utilisons la méthode présentée par [KHA06]**,** qui a été testée sur des documents anciens et a donnée des résultats aussi encourageants que les méthodes précédentes dont voici un tableau comparatif :

Méthode	**Nombre total de mots**	**Nombre de mots reconnus**	**Nombre de mots non reconnus**	**Taux de reconnaissance**
OTSU	2021	1657	364	**81.98 %**
KITTLER	2021	1045	976	**51.70 %**
NIBLACK	2021	1699	322	**84.06 %**
Méthode de [KHA06]	2021	1730	291	**85.60 %**

Tableau 1 comparaison des méthodes de binarisation [KHA06]

Cette méthode à seuillage global calcule le seuil optimal à partir de la luminance et la moyenne des intensités. Souvent la luminance est égale à 255, mais dans la méthode de [KHA06]**,** elle est variable et calculée comme suit: $G_{max}=F_{max}(g)$ où G_{max} est la luminance, et F_{max} est une fonction qui cherche le niveau de gris maximum dans la matrice image.

La moyenne des intensités est donnée par :

$$M=\left(\sum_{y=0}^{dimy}\sum_{x=0}^{dimx} I[x,y]\right) / (dimy \times dimx) \qquad (82)$$

Où : M est la moyenne, dim_x et dim_y dénotent respectivement les dimensions X et Y de l'image, I[X,Y] c'est l'intensité lumineuse.

Soit D = G_{max}-M, la différence entre la luminance et la moyenne.

Le seuil Δ est égale Δ =M-D ou Δ =M-(G_{max}-M) (83)

La matrice image résultante MI[x,y]= $\begin{cases} 0, \text{ si } I[x,y] \leq \Delta. \\ 255, \text{ sinon.} \end{cases}$ (84)

2.3 Sélection des index: dans certaines approche telle que celle de [RAT07] le choix des index se fait automatiquement, cette partie n'est pas incluse dans la notre. La sélection des index est applicable manuellement en faisant intervenir les spécialistes dans chaque domaine afin de choisir les mots les plus pertinents.

Après avoir sélectionner l'index, ceci subira les transformations en niveau de gris et binarisation comme le schéma suivant :

Figure V.4 sélection du mot clé

Ce document contient des versés du Coran. Si le spécialiste décide que le mot الاخرة est le plus représentatif, ce dernier sera sélectionné puis transformé en niveau de gris et binarisé.

2-4 Codage des contours :

La détection des contours dans les images a débutée de façon extrêmement empirique par des opérateurs locaux qui, soit estimaient un gradient, soit transformer l'image par des masques caractéristiques des contours. Dans les années 80, des approches plus systématiques ont été mises en place par Marr et Hildreth, 1980, puis Canny, pour obtenir des contours plus significatifs. Ces travaux ont abouti à une bonne compréhension de ce qu'il faut faire pour

détecter les contours, mais la définition même des contours demeure très vague, ce qui rend ces techniques encore peu efficaces sur un problème concret. [MAI03].

Ici, nous faisons simultanément la détection et le codage du contour. Le code généré est la chaîne de Freeman [HOQ03]**,** qui représente les directions des pixels du contour à 8 voisins.

Les chaînes de Freeman: C'est une technique de représentation des directions du contour (on code la direction le long du contour dans un repère absolu lors du parcours du contour à partir d'une origine donnée). Les directions peuvent se représenter en 4-connexité ou en 8-connexité Le codage d'un contour se fait donc de la façon suivante :

1. transmission des coordonnées absolues du point de départ,
2. transmission de la liste des codes de déplacement d'un point du contour au suivant sur le maillage.

Les codes des contours sont donnés par la figure V.5

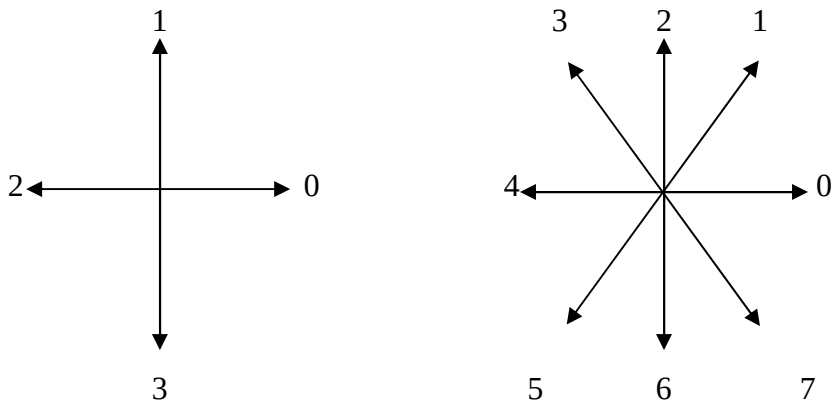
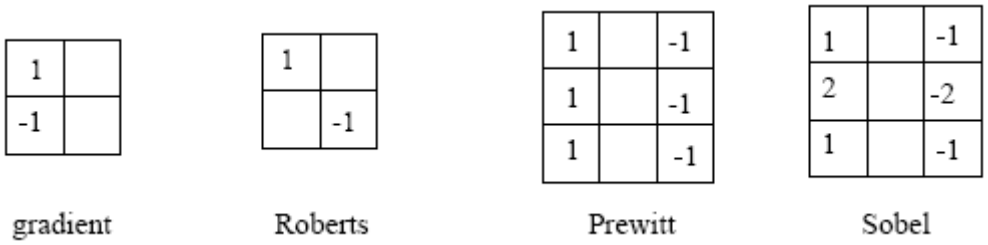


Figure V.5 Les codes de Freeman en 4-connexité (à gauche) et en en 8-connexité (à droite)

Pour détecter les contours, dans un premier temps, nous avons utilisé les masques gradients empiriques proposés à partir d'estimateurs locaux de l'image *f* ou de ses dérivées $\delta f / \delta x$. Ces estimées sont obtenues à l'aide de masques appliqués sur des fenêtres de 2 x 2 pixels ou 3 x 3 pixels. Les filtres les plus utilisés sont, dans l'ordre décroissant Sobel, Roberts, Gradient, et Prewitt:



Les filtres 3 x 3 sont un peu moins précis (c'est-à-dire que les contours qu'ils détectent sont moins bien localisés et souvent épais), mais les images ainsi obtenues sont généralement plus fiables et pe-rmettent des post-traitements plus poussés.

Comme notre objectif n'est pas seulement la détection des contours mais aussi le suivi des directions des pixels, la démarche utilisée la dessus est devenu bloquante, car les contours trouvés sont épais (souvent l'épaisseur est de 3 pixels) la figure V.6 montre le résultat d'une détection de contour de la lettre ا

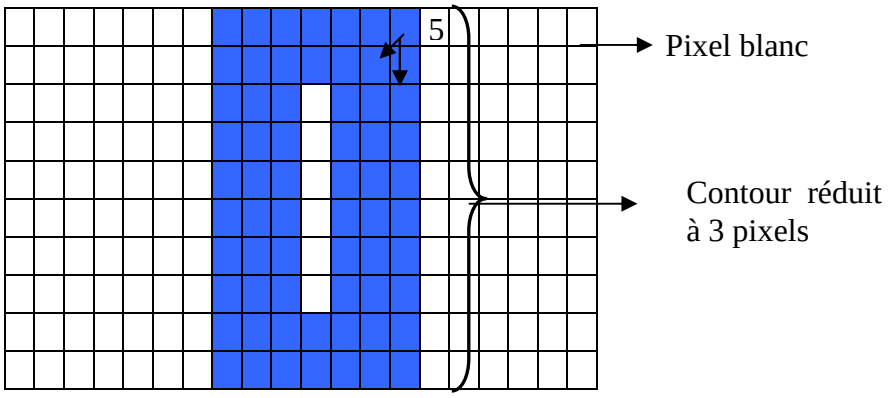


Figure V.6 détection de contour de la lettre ا par filtre de Prewitt

Le codage des contours basé sur le suivi des pixels est irréalisable sur un contour épais, comme le montre l'exemple de la figure. Pour passer du pixels 1 vers le pixels 2 nous avons la choix entre 2 pixels au lieu d'un seul. Pour remédier à ce problème nous avons utilisé la méthode utilisée dans [CHA03] qui permet de tracer un contour mince à l'épaisseur d'un pixel. Le principe de notre méthode est de commencer par le pixel le plus haut puis en suit le cheminement jusqu'à ce qu'on revient au premier pixel. L'algorithme utilisé dans cette démarche est le suivant:

Algorithme de détection du contour :

Début

Largeur_max_parcourue :=0 ; NB_sous_mot=0

1 - Transférer les niveaux de gris de l'image vers une matrice M[i,j] tel que M[i,j]=0 si le pixel est noir et 255 sinon

2 - Trouver le pixel le plus haut à gauche de l'image et mettre M[i,j]:=5

3 - Chercher le prochain pixel en utilisant la formule d:=d+3 mod(8)

4 - si j > Largeur_max_parcourue alors Largeur_max_parcourue :=j

5 - Pour chaque pixel parcouru mettre sa valeur à 1 M[i,j] :=1

6 - si M[i,j] ≠ 5 aller à 3 sinon j= Largeur_max_parcourue ;

7- NB_sous_mot := NB_sous_mot+1 ;

8 - aller à 2

9- pour j=1 à Largeur_max_parcourue si M[i,j] ≠1 ou M[i,j] ≠5 alors M[i,j]=255 ;

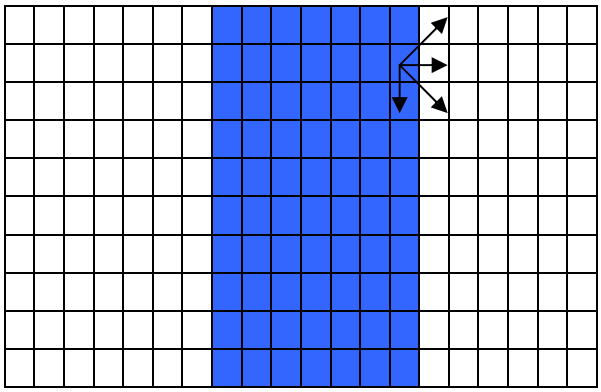
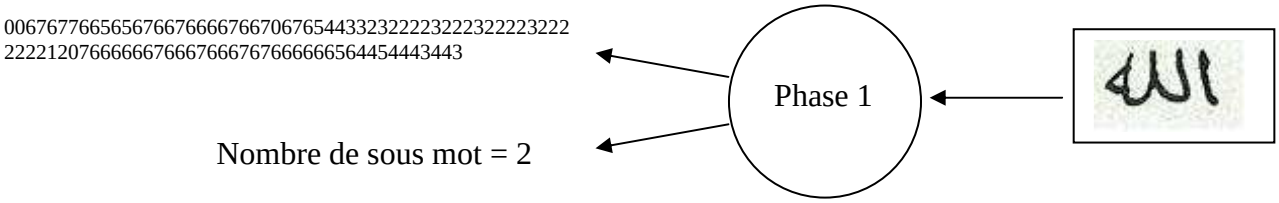


Figure V.7 détection de contour dans l'image originale

Selon l'exemple présenté dans la figure V.7 pour passer du 2ème pixel vers le 3ème prochain pixel testé est défini par d=d+3 mod(8) , l'ancienne valeur de « d » est 6, donc d=6+3 mod(8) = 1. On suit la direction 1 la valeur de m[i,j]=255 (pixel blanc), on teste la direction 0, m[i,j]=255, on teste à nouveau la direction 7 elle donne la même chose et enfin lorsqu'on teste la direction 6 on a m[i,j]=0 , donc on met m[i,j] à 1 et d à 6 et on passe au test suivant. Le résultat final de cet algorithme est le suivi des contours en plus du nombre de composant connexes.



Figue V.8 processus de codage

2.5 Optimisation du code : La chaîne de code obtenue de la première phase n'est pas fiable, puisque deux images du même mot peuvent avoir deux séquences de codes différentes. Ce qui nuit à la phase de recherche. Pour remédier à ce problème nous avons proposé un traitement supplémentaire et primordial, de la chaîne de code. Ce traitement se fait en deux étapes :

1- Etape amincissement : Cette opération consiste à éliminer les pixels éloignés du contour et qui peuvent surgir dans des lignes droites, horizontales ou verticales. Pour ce faire nous avons utilisés la méthode proposée par [NAI08]. La figure V.9 nous montre les différents cas à traiter**.**

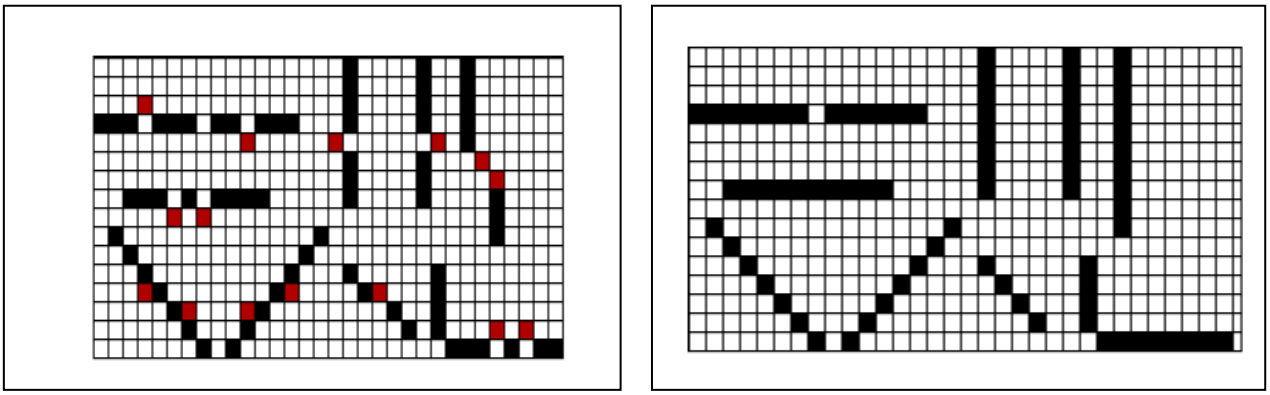


Figure V.9 Différents cas de pixels erronés à éliminer

L'élimination des pixels erronés se fait à l'aide de l'algorithme suivant :

BCi, i=1..n le vecteur qui contient la chaîne de code

1 - si BCi±n(n=1,2) sont semblables, et BCi ≠ BCi±n alors BCi = BCi+1.

2 - si (BCi−n(n=1, Soit 2) = BCi+1 et ≠ BCi) ou (BCi+n(n=1,2 = BCi−1 et ≠ BCi) alors BCi = BCi+1 (ou BCi = BCi−1).

3 - si (BCi−n(n=1,2) = BCi+2 et BCi ≠BCi+2) ou (BCi+n(n=1,2) = BCi−2 et BCi ≠ BCi−2) alors BCi = BCi+2 ou BCi = BCi−2).

Le tableau ci-dessous illustre les différents cas à traiter :

Avant traitement	Après traitement
(0, 7, 1) / (0, 1, 7)	(0, 0, 0)
(2, 3, 1) / (2, 1, 3)	(2, 2, 2)
(4, 5, 3) / (4, 3, 5)	(4, 4, 4)
(6, 5, 7) / (6, 7, 5)	(6, 6, 6)
(1, 2, 0) / (1, 0, 2)	(1, 1)
(3, 4, 2) / (3, 2, 4)	(3, 3)
(5, 6, 4) / (5, 4, 6)	(5, 5)
(7, 0, 6) / (7, 6, 0)	(7, 7)

Tableau 2 différents cas à traiter

Comme c'est montré dans la figure V.9, cette étape consiste à éliminer les pixels éloignés, ces pixels qui sont obtenus durant la phase de détection des contours peuvent fausser l'information sur l'objet recherché. Par exemple la lettre ا peut avoir une forme horizontale ou légèrement oblique, ou bien la partie inférieure de cette lettre peut changer de direction par rapport à la partie supérieure.

2- Etape réduction de la taille du contour: La réduction de la taille du contour vise à minimiser le temps de calcul, et surtout à avoir le même code pour deux mots, semblables, de tailles différentes. Pour ce faire, nous proposons pour chaque contour un seuil L qui est calculé ainsi: L=(longueur_contour)¼, où longueur_contour est le nombre de pixel du contour. Se seuil est appliqué dans la méthode de [NAI08] afin de détecter les changements des angles; chaque angle doit être limité par deux lignes de longueur ≥ L.

Dans notre approche chaque ligne de contour supérieur à L sera réduite à 4 pixels et chaque ligne inférieure à L sera réduite à 1 pixel. Voir figure V.10

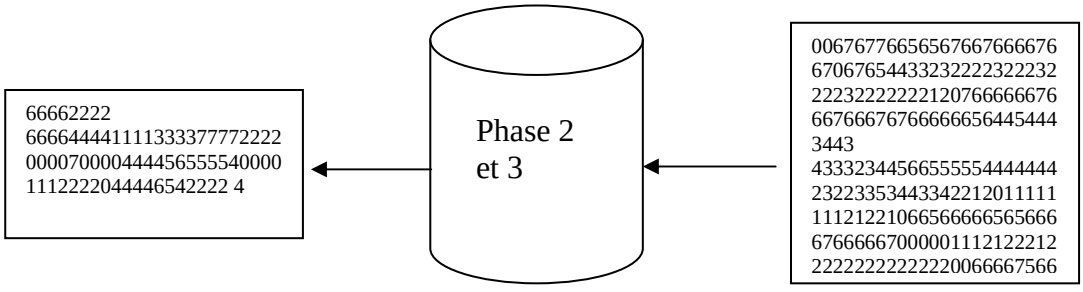


Figure V.10 résultat de la 2ème : code final optimisé

Après avoir généré le code de Freeman ce dernier sera enregistré dans une base de données dont sa structure est définie comme suit :

La première table contient l'image, le code d'indexe et un code pour ce document, ce dernier sera donné par l'utilisateur, une deuxième table contient le code de Freeman, le nombre de sous mots, et un code pour cet indexe qui sert de liaison entre les deux tables. Voir le schéma dans la figure V.11

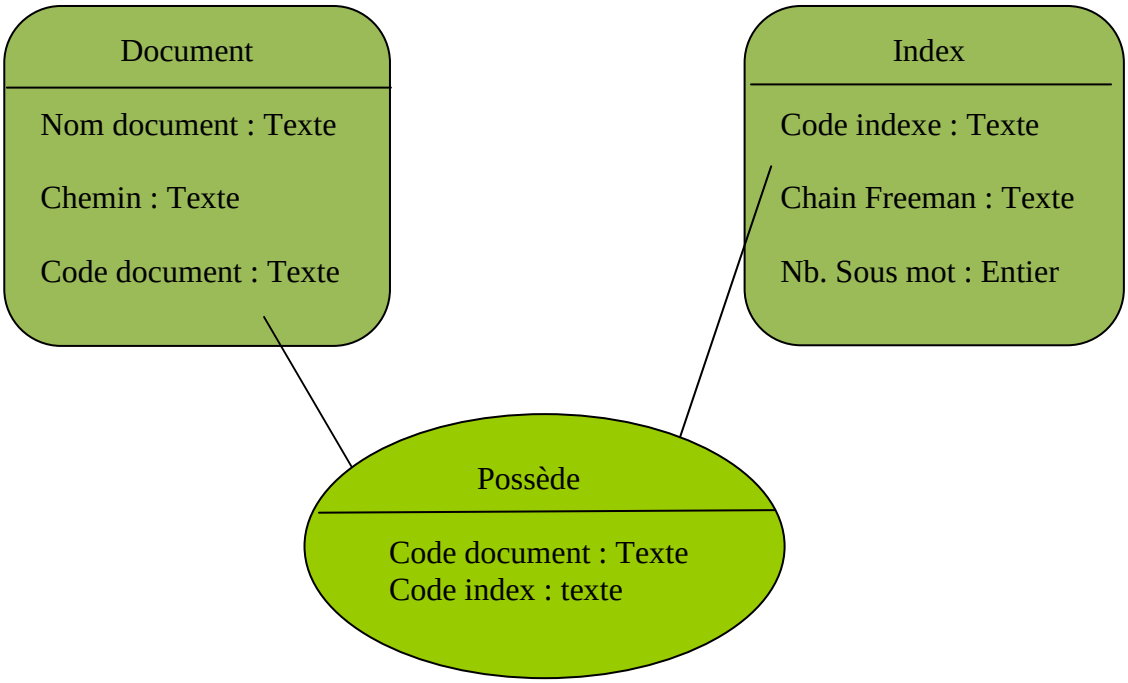


Figure V.11 Schéma de la base de données

3- Recherche : L'utilisateur introduit une image de mot qui passera par une transformation au niveau de gris, une binarisation, puis une détection de contour et extraction du code optimisé et le nombre de sous mots.

Ces deux caractéristiques seront comparés avec la liste des index et stockés dans la table des index. Le facteur le plus important est le nombre de sous mots. Ce facteur minimise le nombre d'index comparés durant cette phase, par exemple le mot صحراء composé de 3 sous mots ne sera pas comparé avec le mot سراب qui est composé de 4 sous mots. Premièrement, le système sélectionne touts les index qui ont le même nombre de sous mots que le mot recherché. Ensuite les chaînes de codes de ces index sélectionnés seront comparées l'une après l'autre avec le code optimal du mot recherché. Et enfin les cinq premiers index, les plus

proches, seront affichés. La comparaison se fait à l'aide de l'algorithme DTW dont voici le pseudo code :

Soit D une matrice de M x N.

1. $D(1,1) = d(x_1,y_1)$
2. pour m = 1 à M
3. $D(m,1) = D(m-1,1) + d(x_m,y_1)$
4. pour n = 1 à N
5. $D(1,n) = D(1,n-1) + d(x_1,y_n)$
6. pour m=2 à M
7. pour n=2 à N
8. $D(m,n) = \min \left\{ \begin{array}{c} D(i, j-1) \\ D(i-1, j) \\ D(i-1, j-1) \end{array} \right\} + d(x_m,y_n)$
9. d(x,y)= 0 si x=y et 1 sinon.
10. La distance est D[M,N].

4. Conclusion: Ce chapitre présente la démarche utilisée pour le développement d'un système d'indexation et de recherche par le contenu dans les documents arabes anciens.

1- Implémentation :

La conception d'un système d'indexation et de recherche d'images nécessite l'utilisation de nombreux outils informatiques. Ce chapitre présente l'architecture et les choix d'implémentation retenus.

Notre système est conçu avec une architecture client / serveur, l'application est en orienté objet et l'outils de développement est Microsoft Visual C Sharp. Cet outil de développement définit des modules qui s'exécutent sur les postes client et la base de données est stockée sur un serveur dédié.

L'avantage du langage C Sharp est qu'il permet, d'une part, d'effectuer des traitements sur les images, de l'autre part, il permet d'interagir avec une base de données. L'architecture du système est la suivante :

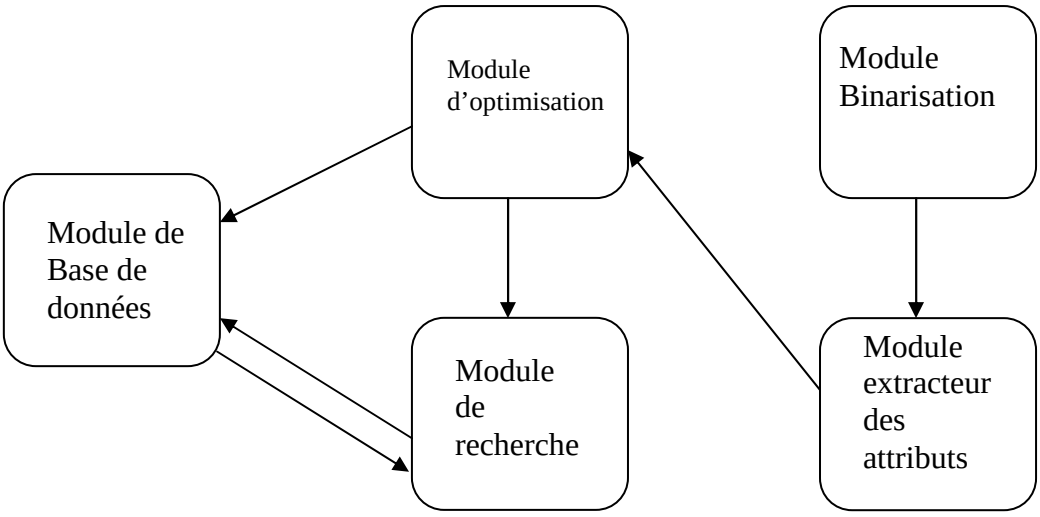


Figure VI.1 : architecture du système d'indexation et de recherche d'image

1.1 Le module binarisation : avant de procéder à la binarisation le système commence par une transformation de l'image couleur vers le niveau de gris, l'image couleur est définie par trois attributs couleur le vert le rouge et le bleu. Pour combiner ces attributs et reproduire un pixel gris nous appliquons la formule suivante :

```
pixel= (0.3*rouge+0.59*vers+0.11*bleu)
```

Chaque pixel gris sera ajouté à la matrice d'image sur laquelle nous effectuons des traitements de binarisation.

La méthode de binarisation appliquée est une méthode à seuillage global, le système commence par un calcul du seuil puis à donner la valeur 0 au pixel dont le niveau de gris est inférieur à ce seuil et 255 autrement.

L'algorithme de binarisation est le suivant :

```
for (int y = 0; y < height; y++)
{
for (int x = 0; x < width; x++)
{
//on récupère le niveau de gris sur 255
            byte pixel = (byte)(.3 * newPixel[0] + .59 * newPixel[1] +
            .11 * newPixel[2]);
i = (pixel >> 2);
//on récupère le niveau de gris maximum
if (i > fmax1) fmax1 = i;
M += i;
newPixel += 3;
}
newPixel += offset;
}
//calcul de la moyenne M

M = M / (height * width);
D = fmax1 - M;
//on calcule le seuil teta

teta = M - D;
//procéder à la binarisation
if ((pixel >> 2) > teta)
newPixel1[0] = newPixel1[1] = newPixel1[2] = 255;
else
newPixel1[0] = newPixel1[1] = newPixel1[2] = 0;
ampl[y][x] = newPixel1[0];
newPixel1 += 3;

bitmap.UnlockBits(bmpData);
```

1.2 Le module extraction des attributs : C'est la deuxième phase qui suit la binarisation. L'information recherchée est les directions des pixels du contour, pour ce faire, nous devons suivre deux étapes :

1- détection du point de départ : l'image est parcourue de droite à gauche et de haut en bas, et le premier pixel noir trouvé (le plus haut à droite) est désigné point de départ exemple : pour la lettre ر ci dessous

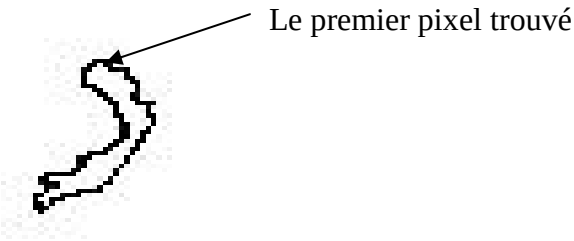


Figure VI.2 détection du premier pixel

2- suivi du contour : le suivi du contour commence à partir du point de départ en suivant la direction des aiguilles de la montre. On parcourt touts les pixels adjacents et en enregistre dans une chaîne de caractères le sens de la direction. L'algorithme de suivi du contour est trop long, pour l'expliquer nous allons utiliser des images et monter juste quelque ligne de code.

0000
0000
0000
0000
0000
0000
0000
0000
0000

Figure VI.3 la lettre ر

Soit la figure VI.3 qui représente la lettre ر, les zéros indiquent les pixels noirs est le reste sont les pixels blancs. La première étape à faire c'est de rechercher le premier pixel le plus haut à droite est de mettre sa valeur à 5 au lieu de zéro voir figure VI.4

0005
0000
0000
0000
0000
0000
0000
0000
0000

Figure VI.4 Marquage du premier pixel

Voici le code source correspondant :

```
// trouve est un boléen initialiser à faux c'est-à-dire pas encore trouver
un pixel noir

if (ampl[x][y] == 0 && trouve == false)
{
ampl[x][y] = 5;
x1 = x;
y1 = y;
trouve = true;
}
```

La deuxième étape est une étape de suivi de contour. Le problème majeur posé dans cette étape c'est à partir de quel direction en cherche le pixel suivant. Dans ce cas on a suivi la formule direcion= (direcion+3) modulo 8. Chaque pixel parcouru et considéré comme un pixel contour son niveau de gris se met à 9 au lieu de 0 et les pixels restant sont mis à 255 (caractère blanc) au lieu de 0, voir figure :

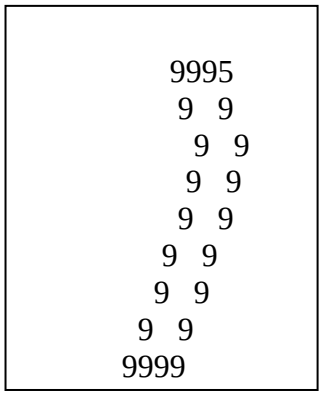


Figure VI.5 Marquage des pixels contours

Le code source de cette étape est trop long on va juste montrer le code qui correspond à la direction 0:

```
switch (d)
{
case 0:
{ if ((x>0 && y>0)&&(ampl[x - 1][y - 1] == 0 || ampl[x - 1][y - 1] == 5 ||
ampl[x - 1][y - 1] == 9))
{
if (cpt != 1) ampl[x][y] = 9;
x--;
y--;

d = 3;
}
else

if ((x > 0)&&(ampl[x - 1][y] == 0 || ampl[x - 1][y] == 5 || ampl[x - 1][y]
== 9))
{
if (cpt != 1) ampl[x][y] = 9;
x--;
```

```
d = 2;
}



else
if ((x > 0)&&(ampl[x - 1][y + 1] == 0 || ampl[x - 1][y + 1] == 5 || ampl[x
- 1][y + 1] == 9))
{
if (cpt != 1) ampl[x][y] = 9;
x--;
y++;

d = 1;
}
else
if ((y<W-1)&&(ampl[x][y + 1] == 0 || ampl[x][y + 1] == 5 || ampl[x][y + 1]
== 9))
{
if         (cpt         !=         1)         ampl[x][y]         =         9;
y++;
;
d = 0;
}
else
if ((x < H-1 && y<W-1)&& (ampl[x + 1][y + 1] == 0 || ampl[x + 1][y + 1] ==
5 || ampl[x + 1][y + 1] == 9))
{
if (cpt != 1) ampl[x][y] = 9;
x++;
y++;
d = 7;
}
else
if ((x<H-1)&&(ampl[x + 1][y] == 0 || ampl[x + 1][y] == 5 || ampl[x + 1][y]
== 9))
{
if (cpt != 1) ampl[x][y] = 9;
x++;
d = 6;
}
else

if ((y > 0 && x<H-1)&&(ampl[x + 1][y - 1] == 0 || ampl[x + 1][y - 1] == 5
|| ampl[x + 1][y - 1] == 9))
{
if (cpt != 1) ampl[x][y] = 9;
y--;
x++;
d = 5;
}
break;

}
```

1.3 Optimisation : lorsqu'on compare deux formes à partir de leurs contours, nous serons en face à deux problèmes majeurs, à savoir, la taille et l'orientation des objets. La phase optimisation de cette approche consiste à réduire la taille des objets. Le principe est simple c'est de jouer sur le code de Freeman en appliquant l'algorithme suivant :

Soit $BC_{i,\ i=1..n}$ le vecteur qui contient la chaîne de code.

1 - si $BC_{i\pm n(n=1,2)}$ sont semblable, et $BC_i \neq BC_{i\pm n}$ alors $BC_i = BC_{i+1}$.

2 - si ($BC_{i-n(n=1,2)} = BC_{i+1}$ et $\neq BC_i$) ou ($BC_{i+n(n=1,2)} = BC_{i-1}$ et $\neq BC_i$) alors $BC_i = BC_{i+1}$ (ou $BC_i = BC_{i-1}$).

3 - si ($BC_{i-n(n=1,2)} = BC_{i+2}$ et $BC_i \neq BC_{i+2}$) ou ($BC_{i+n(n=1,2)} = BC_{i-2}$ et $BC_i \neq BC_{i-2}$) alors $BC_i = BC_{i+2}$ ou $BC_i = BC_{i-2}$).

Exemple : voici les codes de Freeman obtenus pour les lettres ل et ل avant et après optimisation

Avant optimisation :

{0775666667666675765755653544453422322221175576677001011311343112232222222222}

{0063000756770666567167455576666766666766666676666666766666566665555545456354454454353434343452232222222222212221227123317577566566666766767067007011070100127127113122132123213222322222223222232323222221312222122}

La distance est 142

Après optimisation :

{6666542222611112222}

{66665555444422226666111122222}

La distance est 9

On remarque ici que le problème de taille est presque résolu, maintenant on va essayer avec la même lettre mais avec des inclinaisons de 7, 10, 15 et 20 degrés dont voici les codes correspondants :

7 ° : {777766664444222217000022221111} La distance est 17

10° : {077776666555532222577771111222233 3} La distance est 17

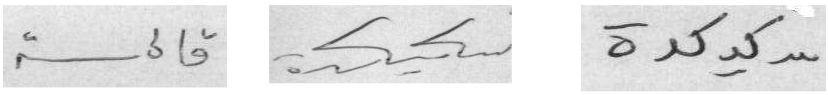
15° : {66665555444422221666671111321222 2} La distance est 14

20° : {000036666555534444121111277776000022222} La distance est 25

L’approche est relativement résistible aux inclinaisons inférieures à 15°.

1.4 Recherche : Le processus de recherche commence par le traitement du mot recherché jusqu’à l’obtention de son code optimal, ensuite ce code sera comparé, à l’aide de l’algorithme de recherche DTW avec touts les codes enregistrés dans la base de données. Pour une recherche rapide le système peut effectuer une présélection selon le nombre de sous mots à l’aide de la clause « SELECT CODE WHERE NB_SOUS_MOT= {NBRE} ». Pratiquement cette présélection diminue les performances du système à cause des coupures, souvent présent dans les manuscrits

Exemple :



.L’algorithme de recherche est le suivant :

```
private void DistanceDeLevenshtein(TextBox Tab1, TextBox Tab2)
{
int Longueur =0;
int Largeur = 0;
int j = 0;
int minimum;
int Distance;
string [] Table1 = new string[400];
string [] Table2 = new string[400];
int [][] Matrice = new int[1][];
for (int i = 0; i < 400; i++)
{
Table1[i] = " ";
Table2[i] = " ";
}

for(int i=0;i<Tab1.TextLength;i++)
{
Table1[i] = Tab1.Text[i].ToString();
}

for (int i = 0; i < Tab2.TextLength; i++)
{
Table2[i] = Tab2.Text[i].ToString();
}
j=0;
Longueur=Tab1.TextLength;
Largeur = Tab2.TextLength;
// reservation de la matrice de distance
Matrice = new int[Longueur+1][];
for (int i = 0; i <= Longueur; i++)
{
Matrice[i] = new int[Largeur+1];
}
// initialisation de Largeur matrice
for (int i = 0; i <= Largeur; i++)
```

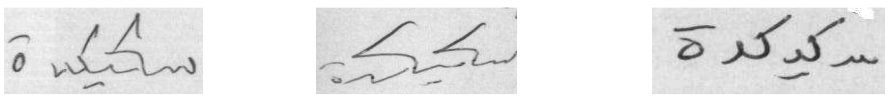
```
{
Matrice[0][i] = i;
}

for (int i = 0; i <= Longueur; i++)
{
Matrice[i][0] = i;
}
int k = 0; int cout = 0;
j = 1;
while (j <= Longueur)
{
k = 1;
while (k <= Largeur)
{
if (Table1[j-1] == Table2[k-1]) cout = 0;
else cout = 1;
minimum = Matrice[j - 1][k] + 1;
if (Matrice[j][k - 1] + 1 < minimum) minimum = Matrice[j][k - 1] + 1;
if (Matrice[j-1][k - 1] + cout < minimum) minimum = Matrice[j-1][k - 1] +
cout;
Matrice[j][k] = minimum;
k++;
}
j++;
}

Distance = Matrice[Longueur][Largeur];
distance.Text = Distance.ToString();
}
```

2- Résultats expérimentaux

Lors d'une première expérimentation menée au labo LRI, nous avons pris un échantillon de lettres postales (environ 1200 enveloppes). Pour indexer et classer ces enveloppes nous avons choisi la ville comme étant l'information la plus pertinente.

Pour chaque index on fait une binarisation, extraction de la chaîne de code et nombre de sous mots, optimisation, puis on lui affecte les documents dont il représente.

Dans cette expérimentation nous avons ignorés le nombre de sous mots car la majorités des mots (index) sont découpés, par exemple pour le même mot Skikda

On a : 

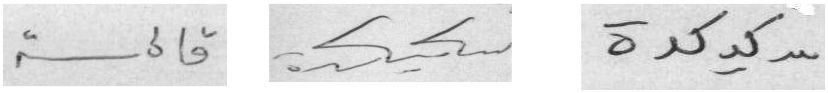
Nous avons testés 500 lettres postales dont 467 uniquement contiennent des adresses valides (avec ville mentionnée). La partie recherche est basée donc sur la comparaison des codes optimisés.

Dans le premier test, on a utilisé presque le quart des mots (100 mots) comme étant des index, et le reste sont réservés pour les tests. Ensuite nous avons augmenté le nombre d'index jusqu'à environ 50% (230 mots). Les résultats obtenus, des deux phases, sont montrés dans le tableau suivant :

Nombre de mots utilisés	Nombre d'index	Nombre de mot testé	Taux de reconnaissance
461	100	361	88.88 %
461	230	461	91.30 %

Tableau VI.1 résultats obtenu selon nombre d'index

Le taux de reconnaissance est proportionnel par rapport au nombre d'index. Les mots qui sont, en général, non détectés sont des mots avec soit une forte inclinaison ou sont découpés. Tels que :



Deuxièmement nous avons projetés nos tests sur la base IFN/ENIT cette base de données contient du matériel pour la formation et le test de la langue arabe dans des logiciels de reconnaissance d'écriture manuscrite. Il y a plus de 2200 images binaires sous forme de manuscrit échantillon de 411 scripteurs. Environ 26000 images de mots binaires ont été isolés à partir des formulaires et enregistrés individuellement pour permettre un accès facile. La description pour chaque mot a été enregistrée dans un fichier dans la base de données. Ce fichier contient des informations sur le mot.
Les caractéristiques de la base IFN ENIT :

1- Contient un ensemble de mots manuscrits désignant des noms des villes tunisiennes environ 26459 noms.

2- Extraites à partir de formulaires remplis par des personnes.

3- Sans restriction pour le style d'écriture
4- Pas de lignes d'écriture ou de boîtes utilisées
5- Divisé en quatre ensembles disjoints.

Nous avons testé 400 mots. Cette fois ci on a utilisé trois quarts des mots (300 mots) comme étant des index, et le reste sont réservés pour les tests. Les résultats obtenus, ont été très encourageants et le taux de reconnaissance est de 91,66 %, un échantillon des résultats des tests est montré dans le tableau suivant :

Mot recherché	1er mot trouvé	2ème mot trouvé	3ème mot trouvé
أكودة	أكودة	أكودة	ذفنة
شماخ	شماخ	شماخ	شماخ
الدخانية	المنزه 6	الدخانية	المنزه 6
المنزه 6	المنزه 6	المترة 6	المنزه 6
نقة	نقة	نقة	ذفنة
رأس الذراع	رأس الذراع	رأس الذراع	رأس الذراع
سيدي الظاهر	سيدي الظاهر	سيدي الظاهر	أكودة
نخال	نخال	نخال	نخال

Tableau VI.2 échantillon des tests sur la base IFN / ENIT

2.1 Interprétation des résultats: le tableau ci-dessus nous montre un échantillon des résultats de recherche de quelques mots, le taux de reconnaissance est élevé mais on peut relever les réserves suivants :

1- le nombre de sous mot est un facteur qui n'aboutit pas à des résultats fiables, car d'après les tests on peut distinguer une différence de ponctuations comme: نقة, نقة et نقة et parfois présence des coupures comme : ذفنة

2- le mot الدخانية à échoué pendant la recherche car on voix clairement que les inclinaisons des caractères sont de différents sens : الدخانية الدخانية

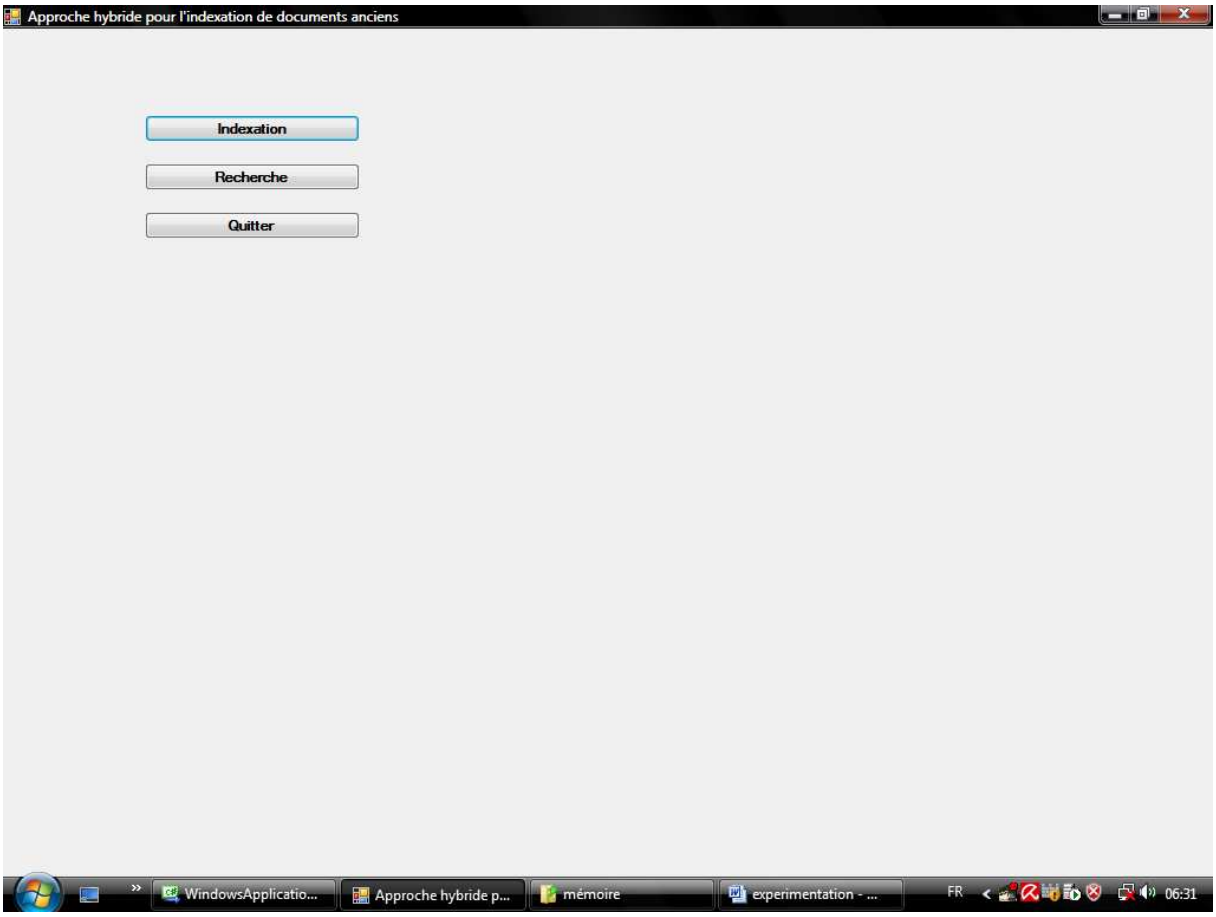


Figure VI.6 Page d'accueil : l'utilisateur peut choisir la rubrique indexation ou recherche

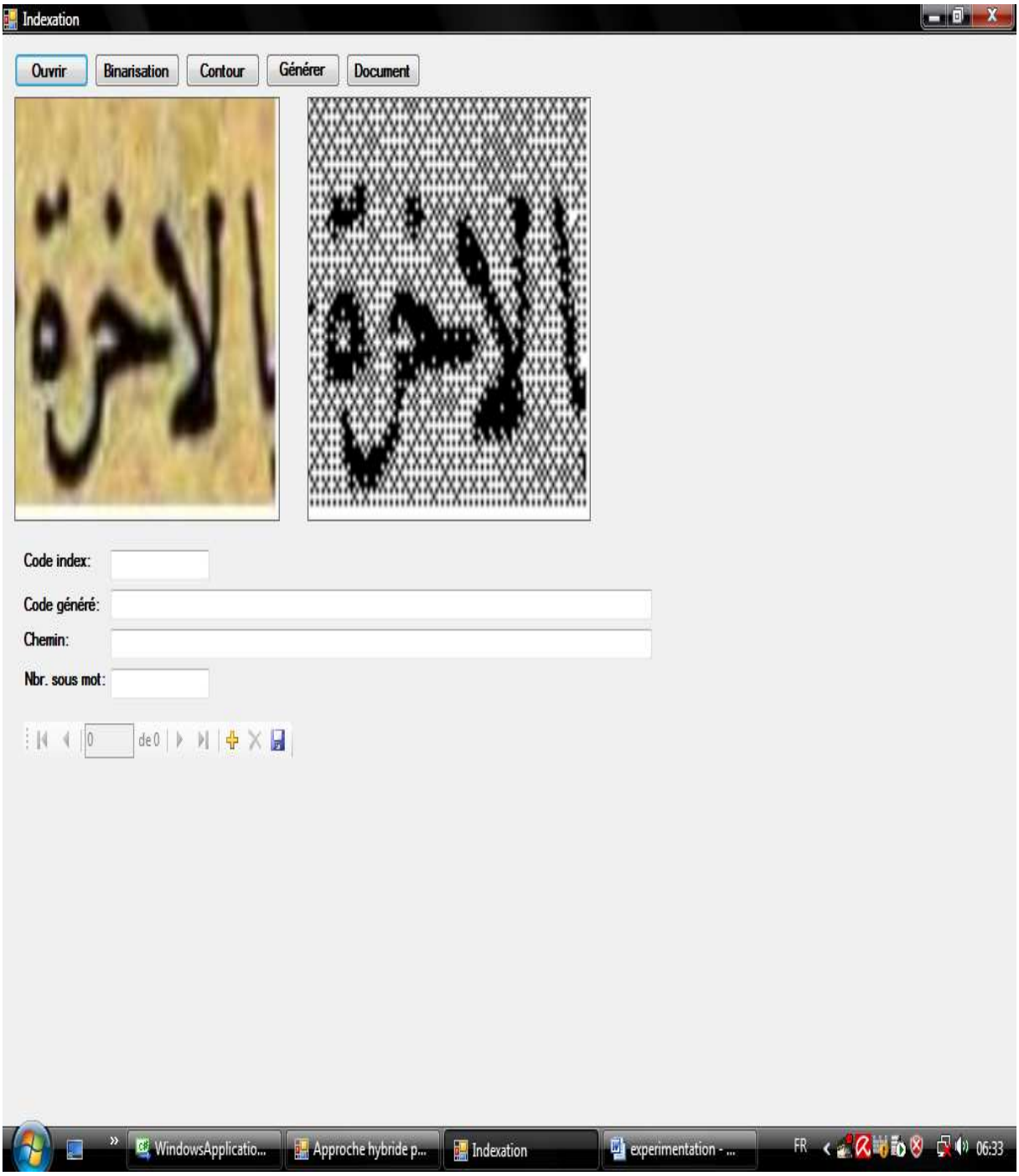


Figure VI.7 Ouvrir un fichier indexe et le transformer vers le niveau de gris

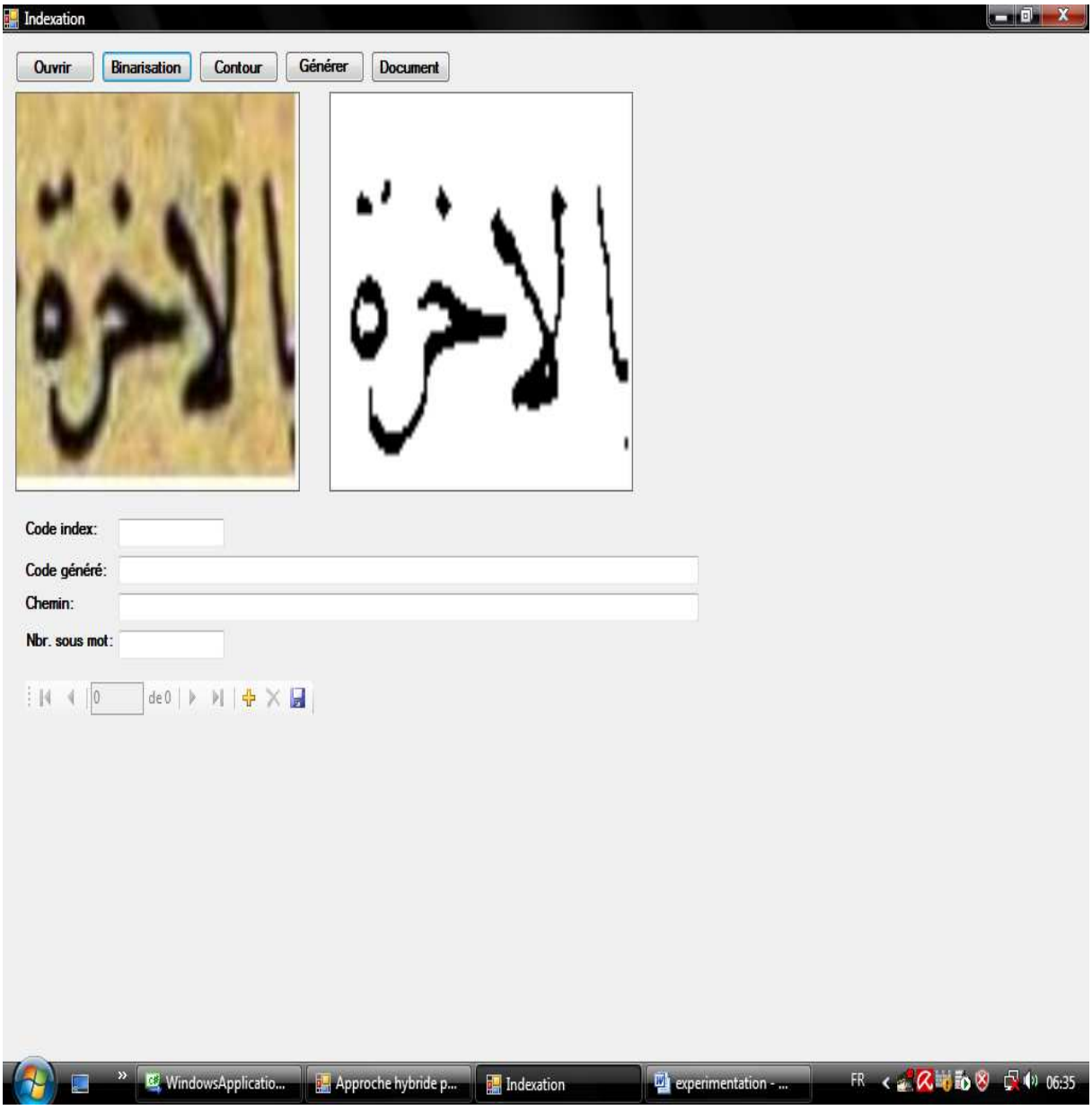


Figure VI.8 Procéder à la binarisation

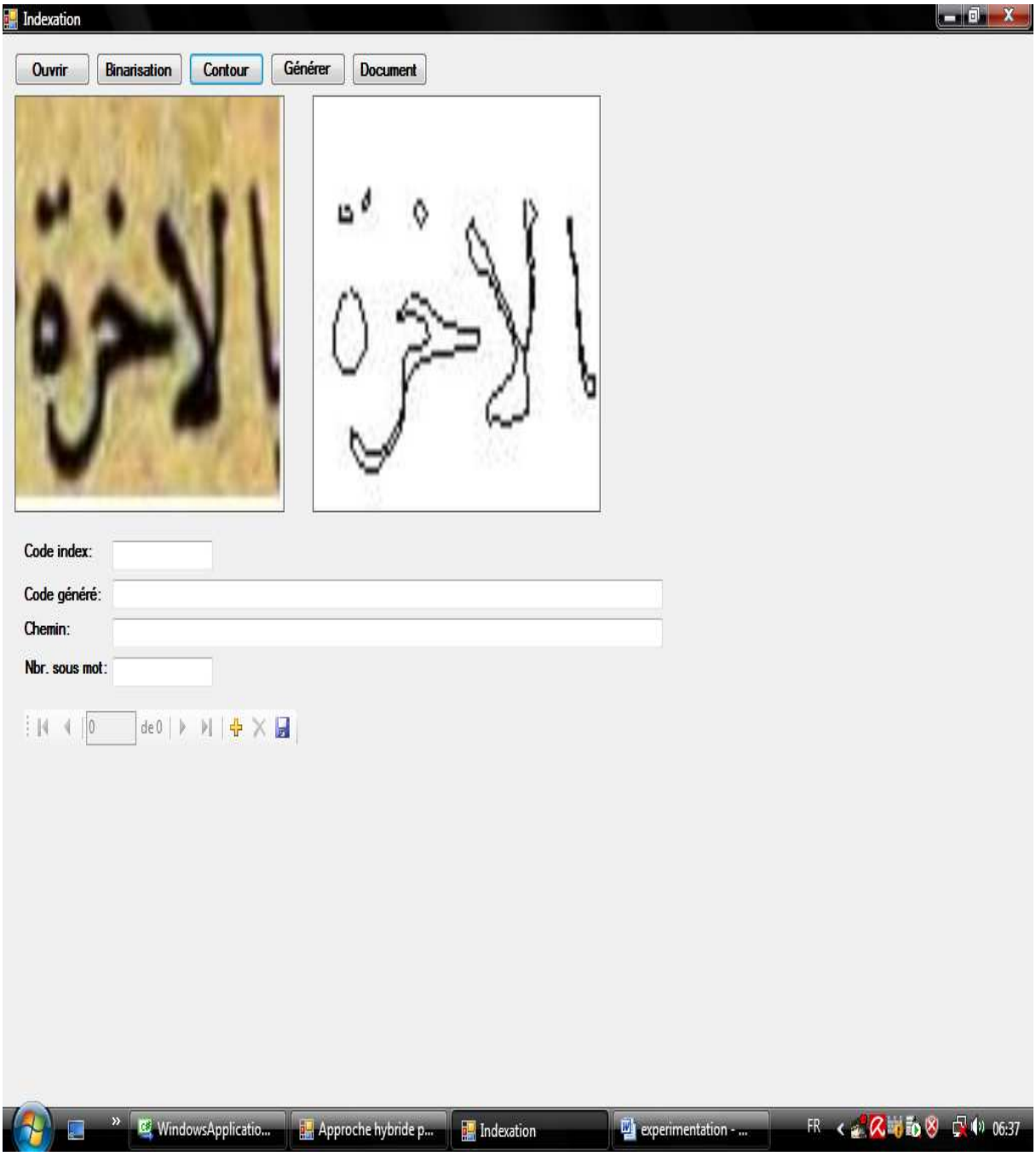
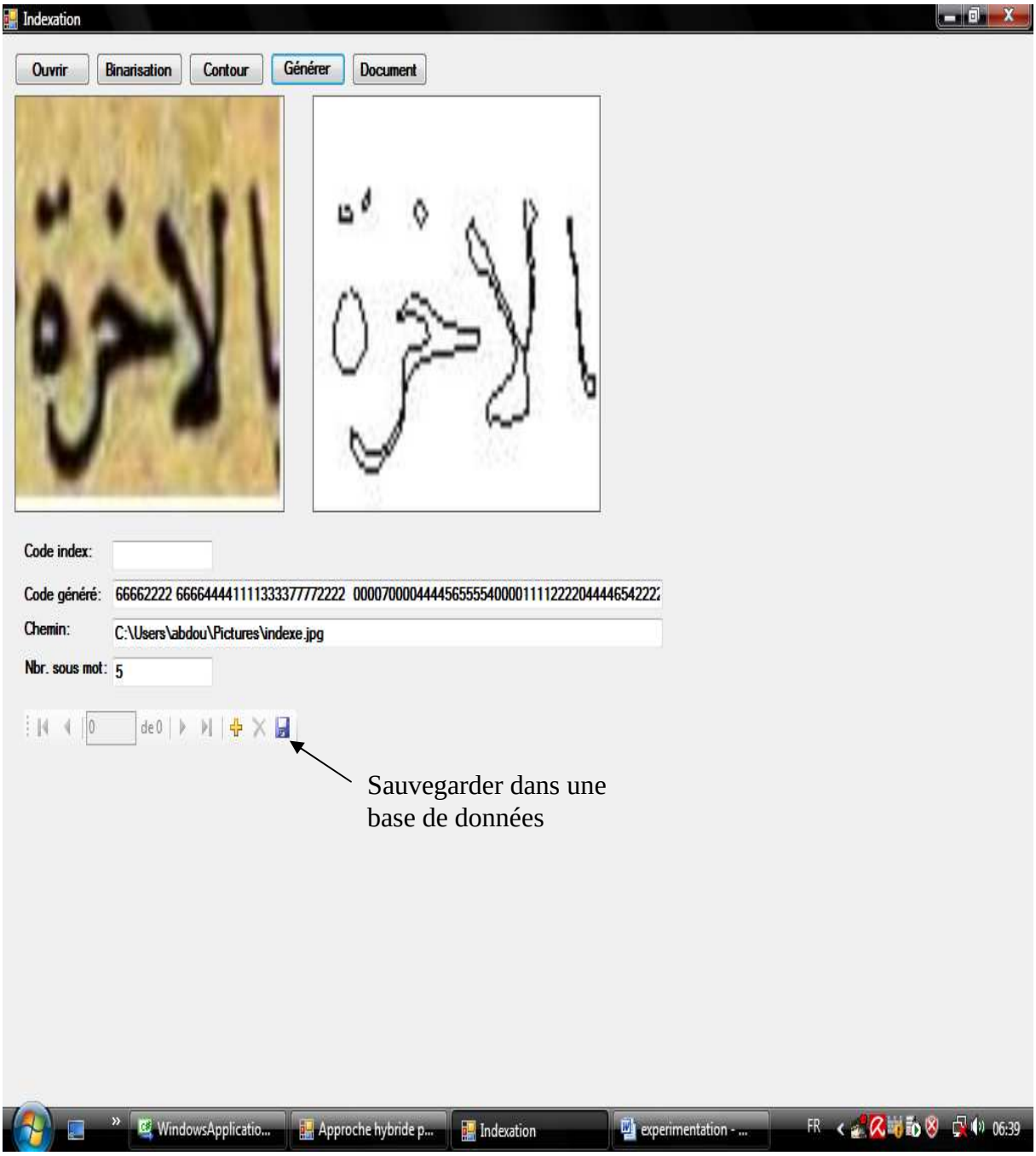


Figure VI.9 détection du contour



4Figure VI.10 Générer code du contour

Figure VI.11 Liaison entre document et indexe

3. Conclusion : les essais nous ont montré que ce système d'indexation et de recherche est quasiment fiable et donne un taux de reconnaissance élevé. Ainsi le système a démontré ses limites devant quelques cas exceptionnels tels que les inclinaisons qui sont supérieures à 15°, et les ruptures dans les écritures. Ces coupures ont rendu inutile le facteur « nombre de sous mots » qui pourra alléger considérablement la phase de recherche. Le problème de déconnection ou de coupure peut être résolu si nous pouvons utiliser la technique proposée dans [ADA07]**,** qui consiste à relier les caractères déconnectés. Aussi pour les inclinaisons causées par erreur d'acquisition des images, d'autres approches permettent de corriger l'angle de dénivellation. Certaines erreurs de reconnaissance peuvent être liées à l'algorithme de recherche utilisé.

Conclusion générale et perspectives:

L’ancien manuscrit est par définition très complexe à analyser, ceci est dû à la non régularité des formes, et leur importante variabilité qui dépend de l'écriture, de la région, de l'auteur, de la période,… etc. Parfois même sur une même page d’un seul auteur. Tenant compte de cette forte variabilité, nous avons choisis d'orienter nos investigations vers une nouvelle approche holistique permettant pour chaque document de détecter les points de contour des formes qu'il contient afin de trouver des similarités partielles sans avoir recours à une segmentation préalable et complète des formes.

Dans la première phase d’indexation on a fait une binarisation à seuillage globale afin d’éliminer le maximum de taches possibles et de récupérer le manuscrit de faible contraste, ce qui augmente considérablement les performances du système. Ensuite, dans la deuxième phase on a introduit un calcul du nombre de sous objet dans le même mot, suivi d’une représentation en chaîne de code des orientations des pixels du contour. La chaîne de code sera ensuite optimisée afin d’éliminer le problème de taille et alléger le système de recherche.

Dans la deuxième partie du projet, à savoir la recherche, le système commence par une élimination des index qui ne possèdent pas le même nombre de sous mot, puis une comparaison des chaînes de codes finaux est appliquée pour trouver les documents correspondants.

L’approche est développée en C# et la base de données Microsoft Access, Les tests sont réalisés sur une base de données composée de noms des villes algériennes et tunisienne. Les résultats obtenus sont très encourageants.

Plusieurs améliorations de notre travail sont possibles :

- Utiliser d’autres techniques de rehaussement des images
- Tester d’autres distances de similarité
- Nous avons testé notre système sur quelques documents anciens, nous proposons de construire une base de données plus volumineuse
- Développer des techniques de description du contour plus robustes aux changements d’orientations et d’échelles telles que les approximations polygonales.

Références bibliographiques :

01- [ADA07] T. Adamek, N. O'Connorn, et A. Smeaton « Word matching using single closed contours for indexing handwritten historical documents » , IJDAR 2007

02- [ALF01] R. Alferez et Y.F. Wang. « Image indexing and retrieval using image-derived, Geometrically and Illumination Invariant Features. » IEEE Multimedia Conference, Japan, 2001.

03- [AME07] Z. Ameur, A. Adane, et S. Ameur « Etude comparative des méthodes de classification appliquée à la segmentation d'images texturées », SETIT 25/03/2007

04- [ARI03] N. ARICA « Shape : representation, description, similarity and recognition » thèse de docteur de l'université de the graduate school of natural and applied sciences , Septembre 2003

05- [BAR92] D. Barba et J. Benois « Image segmentation by region-contour cooperation for image coding » In ICPR92, La Haye, 1992.

06- [BEL02] S. Belongie, J. Malik, et J. Puzicha. « Shape matching and object recognition using shape contexts » IEEE Transactions on Pattern Analysis and Machine Intelligence, Avril 2002.

07- [BEL97] A. Belaïd, « Computer aided design of models of page for their use in recognition of documents » Workshop one Electronic Page Models LAMPE, 1997

08- [BER02] J. Bernd «Digital image processing», Springer 2002

09- [BIA 96] A BIANCARDI et A. MÉRIGOT «Connected Component Support for Image Analysis Programs », International Conference on Pattern Recognition ICPR, 1996.

10- [BOU05] A. Boucher et T. Le « Comment extraire la sémantique d'une image », SETIT 2005

11- [CAN86] F. Canny. «A computational approach to edge detection» PAMI, 1986.

12- [CHA03] F. Chang et C. Chen « A Component-Labeling Algorithm Using Contour Tracing Technique » ICDAR 2003

13- [CHE08] S. CHERALA et P. REGE « Palm leaf manuscript colour document image enhancement by using improved adaptive binarization method », ICVGIP.2008.64

14- [CHR06] F. Christine et R. Noêl « Traitement des images » chapitre II page 374, Dunod 2006

15- [DEL06] M. Delalandre et J. M. Ogier « Un système pour l'indexation rapide d'image de lettrine » , CIFED, 2006

16- [DER87] R Deriche, J. P. COCQUEREZ « segmentation d'images par extraction de composantes connexes basée sur une détection optimale des contours » MARI CESTA PARIS, 1987
17- [DEV82] P.A. Devijver et J.Kittler .J, « A Statistical Approach » Pattern Recognition – Prentice/Hall International, INC.,London. 1982
18- [DRO94] V. DROOGENBROECK « Traitement d'image numériques au moyen d'algorithmes en utilisant la morphologie mathématique et la notion d'objet : application au codage », thèse de docteur de l'école nationale supérieure des mines de Paris, centre de morphologie mathématique 1994.
19- [EGL07] V. Eglin, S. Bres, et C. Rivero « Hermite and Gabor transforms for noise reduction and handwriting classification in ancient manuscripts », IJDAR 2007
20- [ELG05] H. Elghazel, K. Idrissi, C. Ben Amar et A. Baskurt « Approche textuelle pour la recherche d'image » SETIT 2005
21- [FAU03] J. FAUQUEUR, « Contributions pour la Recherche d'Images par Composantes Visuelles » thèse de docteur de l'université de Versailles le 21/11/2003
22- [FER06] V. Fernandez, S. Herranz, R. Unanue, et A. Rubio « Naive Bayes Web Page Classification with HTML Mark-Up Enrichment », ICCGI 2006
23- [FRE86] H. Freeman. « On the encoding of arbitrary geometric configurations » IRE Transactions on Electronic Computers, pages 260_268, 1961.
24- [GAT08] B. Gatos, I. Pratikakis et S.J. Perantonis « Efficient Binarization of Historical and Degraded Document Images », DAS 2008
25- [GUI03] L. Guigues « Modèles Multi-Échelles pour la Segmentation d'Images » thèse de docteur de l'université de Cergy-Pontoise le 12/12/2003
26- [HAR82] A. Hardy. « Une nouvelle approche des problèmes de classification automatique : un modèle - un nouveau critère- des algorithmes- des applications ». Thèse de doctorat, facultés des sciences de Namur, Belgique, 1982.
27- [HOQ03] S. Hoque, K. Sirlantzis, M. C. Fairhurst « A New Chain-code Quantization Approach Enabling High Performance Handwriting Recognition based on Multi-Classifier Schemes », ICDAR 2003
28- [HOR77] S. L. Horowitz et T. Pavlidis « picture segmentation by a direct split and merge precedure » CMETIMALY77 page 101, 1977
29- [JOU07] G. Joutel, V. Eglin, S. Brès et H. Emptoz « Extraction de caractéristiques dans les images par transformée multi-échelle » GRETSI 2007

30- [JOU05] N. Journet, J.Y. Ramel, R. Mullot, et V. Eglin « Analyse des Orientations pour la caractérisation d'Images de Documents de la Renaissance », GRETSI, 2005
31- [KHA06] A. Khashman et B. Sekeroglu, « A Novel Thresholding Method for Text Separation and Document Enhancement », Industrial Technology, 2006. ICIT 2006.
32- [KIT86] Kittler, J. et Illingworth, J. « Minimum Error Thresholding » Pattern Recognition, vol. 19, pp.41-47, 1986
33- [KON07] T. Konidaris, B. Gatos, K. Ntzios, I. Pratikakis, S. Theodoridis et S. J. Perantonis « Keyword-guided word spotting in historical printed documents using synthetic data and user feedback », IJDAR 2007
34- [KUN93] M. Kunt « Traitement numérique des images » presse polytechnique et universitaire romande, éditeur scientifique et technique, vol 2, 1993
35- [LAN05] J. LANDRE «Analyse multi résolution pour la recherche et l'indexation d'images par le contenu dans les bases de données images » thèse de docteur, Université de Bourgogne, 2005.
36- [LEE02] G. Leedham, S. Varma, A. Patankar, et V. Govindaraju, « Separating text and background in degraded documents images–a comparison of global thresholding techniques for multi-stage thresholding », Proceedings of the 8th International Workshop on Frontiers in Handwriting Recognition, Niagara-ontheLake, pp. 244–249, 6–8 August, 2002
37- [LLA07] J. Llados et G. Sanchez « Indexing Historical Documents by Word Shape Signatures », ICDAR 2007
38- [MAC67] J. MacQueen. «Some methods for classification and analysis of multivariate observations». Proc. of the Fifth Berkeley Symp. on Math. Stat. and Prob., 1967.
39- [MAI03] H. Maître, "Le traitement des images", Chapitre 5, Dunod, 2003.
40- [MAR 03] S. Marinai, E. Marino et G. Soda « Indexing and retrieval of words in old documents », ICDAR 2003
41- [NAI08] N. Nain, V. Laxmi, et B. Bhadviya « Corner Detection using Difference Chain Code as Curvature », SITIS 2007
42- [NIB86] W. Niblack « An Introduction to Digital Image Processing » Prentice Hall, pp.115-116. 1986
43- [OTS76] N. Otsu « A Threshold Selection Method from Gray-Level Histogram » IEEE Transactions on Systems, Man, and Cybernetics, vol. 9, pp. 62-66. 1976.
44- [OTS79] N. Otsu, «A thresholding selection method from gray level histogram », IEEE Trans. on Systems, Man and Cybernetics. vol.9, pp. 62-66, March 1979.
45- [OPP95] A.V.Oppenhein et R.W.Schafer, « Dgital Signal Processing », USA, 1995.

46- [OUA99] N. OUARAB, Y. SMARA et J. P. RASSON « Utilisation de méthodes de classification hiérarchique pour une classification supervisée d'images satellitaires », GRETSI le 13/9/1999

47- [RAB86] L.R. Rabiner et B. H. Juang « An introduction to hidden markov models » IEEE ASSP Magazine, pages 4–16, 1986.

48- [RAM07] J. Y. Ramel, S. Leriche, M. L. Demonet et S. Busson « User-driven page layout analysis of historical printed books », IJDAR 2007

49- [RAT07] T. M. Rath et R. Manmatha « Word spotting for historical documents », IJDAR2007

50- [RED07] R. REDONDO « New Contributions on Image Fusion and Compression Based on Space-Frequency Representations » thèse de docteur de l'université polytechnique de Madrid, 2007

51- [SAU97] J. Sauvola, T. Seppänen, S. Haapakoski, et M. Pietikäinen « Adaptive document binarization », Proceedings International Conference on Document Analysis and Recognition, vol. 1, pp. 147–152, 1997

52- [SHA08] J. Shashank, P. Kawshik, K. Srinathan et C. V. Jawahar « Private content based image retrieval » IEEE conférence en Computer Vision and Pattern Recognition, 2008

53- [SHI97] Shi J. et Malik J « Normalized cuts and image segmentation » computer vision and pattern recognition, 1997

54- [SID02] S. SIDHOM « Plate-forme d'analyse morpho-syntaxique pour l'indexation automatique et la recherche d'information : de l'écrit vers la gestion des connaissances », thèse de docteur de l'université de CLAUDE BERNARD – LYON 1 le 11/03/2002

55- [SME00] W.M. Smeulders, M. Worring, S.Santini, A.Gupta et R.Jain. « Content-Based Image Retrieval at the End of the Early Years » . IEEE Trans. Patt. Anal. and Machine Intell., vol. 22, no 12, pp. 1349-1380, Dec.2000

56- [SNO00] S. Snoussi Maddouri, H. Amiri, A. Belaid et N. Nouri « Caractérisation Elliptique par Coefficients de Fourier du Contour du Script Arabe en vue de la Normalisation des Caractères », CIFED, 2000

57- [WRO87] B. WROBEL, O. MONGA « Segmentation d'images naturelles: coopération entre un détecteur-contour et un détecteur-région », GRETSI le 01/06/1987

58- [WHU93] WHU et Leahy « an optical graph theorie approch to data clustering, theory and application to image segmentation » transaction on pattern analysis and machine intelligence, 1993

59- [XIA04] W. Xiaoling et X. Kanglin « A Novel Direction Chain Code-Based Image Retrieval », CIT2004

60- [YAN02] Y. Yang, S. Slattery, et R. Ghani. « A study of approaches to hypertext categorization ». Journal of Intelligent Information Systems, 2002.

61- [BEU90] S. Beucher « Segmentation d'images et morphologie mathématique » thèse de docteur de l'école des Mines, Centre de Morphologie Mathématique. Paris 1990