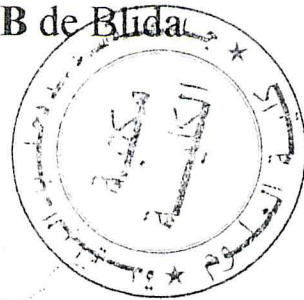


REPUBLIC ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTRE DE L'ENSENGEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

Université SAAD DAHLEB de Blida



Faculté des Sciences

Département Informatique

Mémoire Pour L'obtention

Du Diplôme De Master En Informatique

Thème

ORGANISATION D'UNE BASE
D'IMAGES EN ARBRE
QUATERNAIRE GNERIQUE

Présenté par :

DJERRAH Abd_Nacer

Promoteurs :

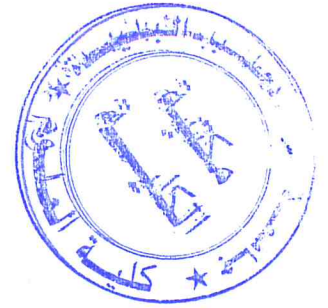
Mr. Amine CHERIF ZAHAR

Mr. Khalfi Ali

Année Universitaire 2012/2013

MA-004-178-1

Remerciements



Tout d'abord, je remercie le bon dieu de m'avoir donné la force et le courage pour arriver là où j'en suis aujourd'hui.

Je remercie en second lieu ma chère famille qui m'a soutenu et encouragé.

Je remercie également mon promoteur Monsieur **Cherif Zahar** et mon

Co-promotrice **Khalfi Ali** pour l'aide et les conseils qu'ils m'ont apporté lors des différents suivis.

Je suis sensible à l'honneur que fait la présence des membres de jury. Qu'ils veuillent accepter mon profond respect.

Dédicace

En signe de respect et de reconnaissance, je dédie ce modeste travail à toutes les personnes qui m'ont toujours aimée, encouragé et aidé tout le long de mon travail.

Des personnes chères que je cite respectivement :

A mes très chers parents qui m'ont toujours soutenu particulièrement durant toutes mes années d'études.

A mes très chers frères et sœurs que j'aime énormément.

A mon Ex binôme Sarah Bouguena.

A tous mes ami(e)s de la fac et tous ceux que j'ai connus auparavant, je précise :

Abd-wahab, Fateh, Ayoub, Zino, Nadia , Housin, Ali, Hamza, Hasan.

Nacer

Résumé

Les images numérisées sont utilisées dans de nombreuses applications géographiques, médicales ou éducatives. Ces applications, et notamment les ateliers de traitement d'images, sont amenées à générer, gérer et stocker des images similaires ou peu différentes, qui occupent beaucoup de place en mémoire. Aussi, des outils efficaces doivent-ils être développés pour stocker et gérer ces images. Les applications gérant des images similaires doivent permettre, en plus du stockage et de l'affichage, de retrouver, modifier, et comparer ces images. Dans ce mémoire, nous avons présenté une structure de données appelée Arbre Quaternaire Générique qui permet de organiser une base d'images avec un facteur de similarité. Puis nous avons proposé un mécanisme de recherche d'image par contenu basé sur notre arbre quaternaire générique.

Mots clef : Arbre Quaternaire, Indexation d'image par contenu, comparaison d'images.

Abstract

The digitized images are used in many geographic applications, medical or educational purposes. These applications, including image processing workshops are brought to generate, manage and store similar or slightly different pictures, which occupy a lot of memory space. Also, effective tools need to be developed to store and manage these images. The applications manager similar images should allow more storage and display, find, edit, and compare these images. In this paper, we presented a data structure called Generic Tree Quaternary who help organize a database of images with a similarity factor. Then we proposed a mechanism contained image retrieval based on our generic quadtree.

Keywords: Tree Quaternary, Indexing image contained, image comparison.

Sommaire

Introduction générale.....	1
Chapitre 1 : Indexation et recherche d'image par le contenu.....	3
I. Introduction.....	3
II. Domaines d'application de l'indexation et la recherche d'images.....	3
III. Indexation.....	4
III.1. Définition.....	4
III.2. Objectif.....	4
III.3. Les enjeux de l'indexation.....	5
III.4. Méthodologie et architecture.....	5
IV. Les phases d'indexation.....	6
IV.1. Indexation logique.....	6
IV.2. Indexation physique.....	6
V. Les méthodes pour indexation et recherche par le contenu.....	7
V.1. Description globale de l'image.....	7
V.2. Description locale de l'image.....	7
V.3. Descripteurs spécifiques.....	8
V.1.1. Description globale.....	8
V.2.1. Description locale.....	10
V.3.1. Description spécifiques.....	10
VI. Problèmes liés à l'indexation.....	11
VII. Similarité dans le domaine des images.....	11
VIII. Systèmes de recherche d'images 12.....	12
VIII.1. Introduction.....	12
VIII.2. Principe.....	13
VIII.3. Exemple de système de recherche d'images (SRI).....	14

VIII.3.1. Le système QBIC	14
VIII.3.2. Le système IKONA	16
Conclusion.....	17
Chapitre 2 : La structure de l'Arbre Quaternaire Générique.....	19
I. Introduction	19
II. L'Arbre Quaternaire Générique	21
II.1. Définition	21
II.2. Principes de l'Arbre Quaternaire Générique	21
II.2.1. Le concept de partage entre les arbres quaternaires	21
II.2.2. Arbre Quaternaire d'Image	22
II.2.3. Similarité entre images	24
II.2.4. Arbre Quaternaire Générique SPI	26
II.2.5. Nœuds génériques	27
II.2.6. L'arbre d'images	28
II.2.7. Graphe de partage et déduction de l'Arbre d'Images	28
II.2.8. Construction de l'Arbre Quaternaire Générique final	33
Conclusion	34
Chapitre 3 : Conception.....	36
I. Introduction	36
II. Les images manipulées	36
III. Construire l'arbre d'images	36
III.1. Les images en noir et blanc	36
III.2. Les images en couleur	49
IV. Recherche des images similaires dans l'arbre d'images	61
Conclusion	61
Chapitre 4 : Implémentation et résultats expérimentaux.....	63
I. Introduction	63
II. Présentation du logiciel.....	63
III. Les Teste	69

III.1 Les images noir et blanc	69
a) Stockage d'image	69
b) Recherche d'une image dans l'arbre d'images	71
III.1 Les images couleur	73
CONCLUSION.....	76
Conclusion générale.....	77
Webographies.....	78

Liste des figures

- Figure 1.1** Les phase off-line, on-line.
- Figure 2.1** Numérotation des quadrants.
- Figure 2.2** Représentation d'une image en arbre quaternaire.
- Figure 2.3** Le Calcul du coefficient de Q-similarité des images a et b.
- Figure 2.4** La représentation de quatre images en arbre quaternaire (Manouvrier, 2000, Figure 5.13).
- Figure 2.5** L'Arbre Quaternaire Générique SPI des images de la Figure 2.4 (Manouvrier, 2000).
- Figure 2.6** L'Arbre Quaternaire Générique SPI des images de la Figure 2.4 (Manouvrier, 2000).
- Figure 2.7** Le graphe de partage de L'Arbre SPI de la Figure 2.6 (Manouvrier, 2000).
- Figure 2.9** Graphe de partage.
- Figure 2.10** Les différent étapes pour construire l'arbre couvrant maximal.
- Figure 2.11** L'arbre couvrant maximum pour le graphe dans la Figure 2.7 (Manouvrier, 2000).
- Figure 2.12** Différent Arbres d'Images possibles (Manouvrier, 2000).
- Figure 2.13** L'arbre quaternaire générique final (Manouvrier, 2000).
- Figure 3.1** Processus de construction de l'arbre générique pour les images en noir et Blanc.
- Figure 3.2** Région d'une image.
- Figure 3.3** Deux régions différentes ont le même pourcentage des couleurs.
- Figure 3.4** Région d'une image.
- Figure 3.5** Représentation d'une image en arbre quaternaire.
- Figure 3.6** Continue de fichier stockage.
- Figure 3.7** Fichier (.aqn).
- Figure 3.8** Affichage de l'arbre et l'image.
- Figure 3.9** Ensemble des images.

- Figure 3.10** Présentation de l'arbre quaternaire générique SPI.
- Figure 3.11** Graphe de partage.
- Figure 3.12** L'arbre couvrant minimum de la figure 3.10.
- Figure 3.13** L'arbre possible pour la Figure 3.12.
- Figure 3.14** Fichier de stockage de l'arbre d'images.
- Figure 3.15** L'arbre quaternaire générique final.
- Figure 3.16** Image en couleur composé de trois couches
- Figure 3.17** Découpage de l'intervalle.
- Figure 3.18** Processus de construction de l'arbre générique pour les images couleur.
- Figure 3.19** Ensemble des images.
- Figure 3.20** Présentation de l'arbre quaternaire générique SPI.
- Figure 3.21** Graphe de partage.
- Figure 3.22** L'arbre couvrant minimum de la figure 3.20.
- Figure 3.23** L'arbre possible pour la Figure 3.22.
- Figure 3.24** Fichier de stockage de l'arbre d'images.
- Figure 3.25** L'arbre quaternaire générique final.
- Figure 4.1** Fenêtre principale.
- Figure 4.2** Menu>Fichier.
- Figure 4.3** Menu/Edit.
- Figure 4.4** Menu/Afficher.
- Figure 4.5** Menu/Paramètre.
- Figure 4.6** Menu/Parametre/parametres globale.
- Figure 4.7** Menu/requête.
- Figure 4.8** Fenêtre de requête.
- Figure 4.9** Menu/langage.
- Figure 4.10** Menu/Info.
- Figure 4.11** JcomboBox de mode.
- Figure 4.12** Les différents paramètres de couleur dominante de mode noir et blanc.
- Figure 4.13** Les différents paramètres de homogénéité de mode noir et blanc.
- Figure 4.14** Les différents paramètres couleur dominante de mode noir et blanc.

Figure 4.15 Les différents paramètres homogénéité de mode couleur.

Figure 4.16 Image noir et blanc.

Figure 4.17 L'affichage de fichier (.aqr).

Figure 4.18 Image noir et blanc.

Figure 4.19 L'affichage de fichier (.aqr).

Figure 4.20 L'affichage de fichier (.aqr).

Figure 4.21 Base d'images.

Figure 4.22 Image requête.

Figure 4.23 Résultat de recherche.

Figure 4.24 Base d'images couleur.

Figure 4.25 Paramètres utilisés.

Figure 4.26 Image requête.

Figure 4.27 Résultat de recherche.

Figure 4.28 paramètres de recherche.

Figure 4.29 Résultat de recherche.

Figure 4.30 paramètres de recherche.

Figure 4.31 Résultat de recherche.

Liste des tableaux

Tableau 2.1 Liste ordonnée.

Liste des abbreviations

MPEG: Moving Picture Experts Group.

RGB: Red, Green, Blue.

RVB: Rouge, Vert, Bleu.

TSV : Teinte Saturation Valeur.

HSV : Hue saturation value.

SRI : Système de Recherche d'Images.

QBIC: query by image content.

IBM : International Business Machines.

INTRODUCTION GENERALE

L'indexation des images par le contenu est un objectif ambitieux qui est rendu possible par les capacités techniques offertes par les matériels actuels, et aussi par les progrès scientifiques du traitement d'images et de la vision par ordinateur.

L'indexation automatique d'images directement à partir de leur contenu numérique est une des solutions possibles et prometteuses pour gérer efficacement les bases de données d'images numériques.

L'interrogation d'une base d'images peut s'appuyer sur des mesures de similarité entre les caractéristiques bas niveau d'une image requête proposée comme exemple, et le résultat attendu est celles des images présentes dans la base.

L'arbre quaternaire Générique rend faisable ou facilite plusieurs opérations n'existant pas dans d'autres structures. Cette structure est donc très utile dans des applications où il est nécessaire de travailler sur une image ou sur plusieurs images simultanément.

Elle doit non seulement optimiser le stockage des images mais également permettre aux utilisateurs d'appliquer des traitements sur les images ou de les comparer.

Dans ce contexte l'objectif de travail consiste à organiser une base d'image stocké en arbre quaternaire générique (AQG).

CHAPITRE 1
Indexation et recherche d'image
par le contenu

Indexation et recherche d'image par le contenu

I. Introduction

Les premiers systèmes de recherche d'images utilisaient des mots-clés associés aux images pour les caractériser. Grâce à cette association de mots-clés, il suffit d'utiliser les méthodes basées sur le texte pour retrouver les images contenant les mots-clés. Plusieurs moteurs de recherche proposent ces recherches d'images basées sur le texte. Ils s'appuient sur le principe simple que dans une page web, il y a une forte corrélation entre le texte et les images présentes.

Le principal problème de ces recherches par mots-clés est que le résultat peut être complètement hors sujet.

Pour cela, des méthodes d'indexation par le contenu, de recherche interactive et de navigation dans des bases d'images sont développées afin de pouvoir les interroger par le contenu, d'une manière ergonomique et intuitive pour l'utilisateur.

II. Domaines d'application de l'indexation et la recherche d'images

Parmi les nombreuses applications de l'indexation et de la recherche d'images par le contenu, on peut citer :

- Les applications scientifiques (ex : dans le cadre de l'imagerie médicales, retrouver les images pathologiques).
- Les applications grand public (ex : commerce électronique).
- L'authentification (ex : détecter la contrefaçon, identifier un visage)
- L'art, l'éducation (ex : recherche encyclopédique d'illustration).
- Les télécommunications (ex : coder et transmettre les images par leur contenu, cf. MPEG-4 et MPEG-7).
- Le design, la publicité (ex : Rechercher une texture spécifique pour l'industrie textile illustrer une publicité par une image adéquate).
- L'audiovisuel (ex : recherche un plan spécifique d'un film).

III. Indexation

III.1. Définition

Indexer une image, c'est rechercher une représentation compacte (moins de données plus de sémantique), significative (relativement au contenu de l'image et aux utilisateurs de la base), rapide à calculer et à comparer.

Donc l'indexation d'une image constitue une des clés importantes pour répondre au problème de la consultation et de la recherche.

III.2. Objectif

Le but de l'indexation d'un document image est d'extraire et de représenter le contenu nécessaire et suffisant pour qu'il soit retrouvé par un utilisateur. Cette indexation se base donc sur une représentation (supportée par un modèle) et sur un processus d'extraction. Afin d'éviter d'extraire des informations non pertinentes dans un contexte donné, chacun des éléments précédents doit intégrer d'une manière ou d'une autre les besoins des utilisateurs. Le but de l'indexation est toujours de créer des regroupements de documents sur un même sujet, mais la description est plus précise.

Une des clés de l'indexation efficace est l'identification de caractéristiques primaires en accord avec le type et le but des recherches visées par le système.

III.3. Les enjeux de l'indexation

Pour s'intéresser à l'indexation et la recherche d'images dans un contexte applicatif, il est commode de subdiviser les bases d'images en deux catégories :

- La première catégorie est celle des bases avec vérité terrain. Les images de ces bases ont le plus souvent un contenu sémantique homogène (visages, mammographies, images satellitaires...).

Les applications visées sont généralement professionnelles. Pour indexer de telles images, le concepteur devra intégrer la vérité terrain définie par l'expert du domaine pour développer un algorithme spécifique, l'objectif étant d'optimiser l'efficacité du système, c'est-à-dire : sa capacité à répondre aussi bien que l'expert.

La recherche d'images dans telles bases est en fait un problème spécifique de reconnaissance (ex : de visage, de tumeurs, etc..).

- La deuxième catégorie regroupe les bases d'images généralement hétérogènes et sans vérité terrain. Ces images concernent généralement le grand public (ex : recherche d'images sur le réseau Internet).

Cette fois l'objectif du système est d'aider l'utilisateur à naviguer intelligemment dans cette base en s'adaptant aux besoins subjectifs de chaque utilisateur.

Le système de recherche doit être aussi flexible que possible, par exemple en proposant à l'utilisateur d'affiner sa requête.

Dans les deux cas, un système de recherche d'images doit permettre à l'utilisateur d'interroger ces bases.

III.4. Méthodologie et architecture

L'indexation des images se divise en deux étapes :

1. Choix d'un espace de représentation des images. Pour représenter le contenu d'une image en machine, on vise une représentation compacte, significative, rapide à calculer et à comparer.

2. Construction d'index c'est à dire stockage efficace des signatures en machine, dans le but de minimiser ultérieurement le coût de la requête.

Après la phase d'indexation, l'objectif du système est de permettre à l'utilisateur de formuler interactivement ses requêtes. C'est lors de cette phase que la similarité entre deux images doit être définie.

Notons que le système doit également offrir la possibilité d'effectuer des requêtes partielles, c'est à dire retrouver les images ayant certains objet (région) similaire à ceux de l'image requête.

IV. Les phases d'indexation

Un système d'indexation comprend généralement deux phases de traitement

IV.1. Indexation logique

L'indexation logique consiste à extraire et à modéliser les caractéristiques de l'image qui sont principalement la forme, la couleur et la texture. Chacune de ces caractéristiques pouvant être considérée pour l'image entière ou pour une région de l'image [5].

IV.2. Indexation physique

L'indexation physique consiste à déterminer une structure efficace d'accès aux données pour trouver rapidement une information. De nombreuses techniques basées sur des arbres (arbre-B, arbre -R, **arbre quaternaires**,....) ont été proposées.

Pour qu'un système de recherche d'images soit performant, il faut que l'indexation logique soit pertinente et que l'indexation physique permette un accès rapide aux documents recherchés [5].

Phases indexation (off-line,on-line)

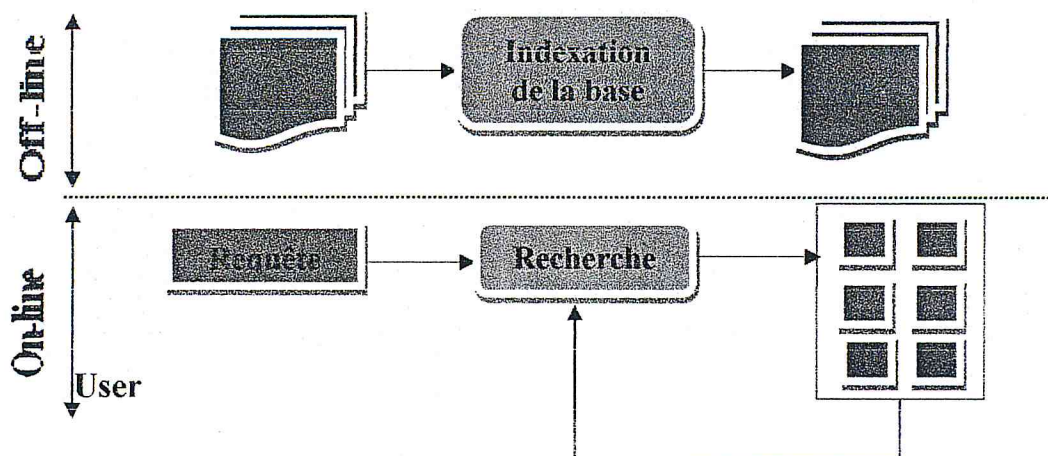


Figure 1.1 –Les phases off-line, on-line

La phase *offline* : Pour la constitution des index de chaque objet (région, image...).

La phase *online* : de définition de la requête et de recherche (dans la base) des objets correspondants au mieux à cette dernière (figure 1.1).

V. Les méthodes pour indexation et recherche par le contenu

Pour faire l'indexation et la recherche d'images, on extrait automatiquement des caractéristiques d'image et les stockées dans un vecteur numérique appelé Descripteur visuel. Il y a 3 types de descripteurs :

V.1. Description globale de l'image

On va faire la recherche approximative sur toute l'image. Pour résoudre ce problème, on indexe les images par les caractéristiques symboliques comme la couleur, la texture, la forme,...

V.2. Description locale de l'image

On va faire la recherche précise sur une partie de l'image. Pour résoudre ce problème, on utilise des modèles pour la reconnaissance d'objets ou il n'y a pas de modèles comme segmentation en régions, détection de points d'intérêt,...

V.1. Descripteurs spécifiques

(Essentiellement biométrie). On utilise des caractéristiques spécifiques des objets, par exemple : les caractéristiques spécifiques de visages,...

V.1.1. Description globale

La couleur

La couleur est le premier descripteur qui est utilisé pour la recherche d'images. Il y a certains de travaux qui ont déjà montré que la couleur est descripteur efficace. Il existe de nombreuses possibilités d'attributs pour caractériser la couleur : l'histogramme, les moments couleur... Avec la même méthode, si on change l'espace de couleur, il peut donner des informations différentes de l'image.

L'histogramme

La signature la plus courante pour traduire l'information de couleur est l'intersection d'histogramme [Swa 91]. Les histogrammes sont faciles et rapides à calculer, et robustes à la rotation et à la translation.

Les espaces de couleur

La deuxième approche essaie de trouver d'autres espaces de couleurs qui se basent sur la perception de couleur de l'humain. L'espace RGB est standard pour les images numériques. Dans cet espace, chaque élément (rouge, vert et bleu) a la même quantité d'information. En général, l'histogramme RGB est fortement sensible à des différences de luminosité entre deux images semblables, car les trois dimensions de l'espace sont affectées.

L'espace TSV (Teinte Saturation Valeur - en anglais HSV) est le plus utile pour la segmentation et la reconnaissance et il a été prouvé un espace très fort dans le système de recherche d'images. Parce qu'il est une représentation plus physique de la couleur. Dans cet espace, on peut séparer pour un pixel : l'intensité du pixel (valeur) et la couleur du pixel (teinte et saturation).

Les moments statistiques

La méthode d'histogramme utilise la distribution complète de la couleur. Les données stockées sont beaucoup. Cela nous met beaucoup de temps et aussi de mémoire. Donc pour résoudre ce problème, au lieu de calculer la distribution complète, dans les systèmes de recherche d'images, on calcule seulement des dominantes caractéristiques de couleur tel que l'espérance, la variance et d'autres moments. Dans [Str 95] les auteurs ont prouvé que les méthodes utilisées des moments statistiques marchent plus vite et donnent des résultats meilleurs que les méthodes d'histogrammes. On peut calculer l'espérance, la variance, les moments sur chaque composante couleur par la formule suivante :

$$E = \frac{1}{N} \sum_{j=1}^N p_j$$

$$\delta = \left(\frac{1}{N} \sum_{j=1}^N (p_j - E)^2 \right)^{\frac{1}{2}}$$

$$s = \left(\frac{1}{N} \sum_{j=1}^N (p_j - E)^3 \right)^{\frac{1}{3}}$$

Ou E est l'espérance, δ est la variance, s est le troisième moment.

La texture

Une texture se caractérise par la répétition d'un motif ou de quelques éléments. Il existe plusieurs méthodes pour analyser la texture.

L'approche de [Har 73] propose une analyse des textures qui cherche à concilier deux approches du problème Il produit les matrices de cooccurrences.

La forme

Des caractéristiques de forme sont extraites à partir des régions dans les images segmentées. Des formes peuvent être représentées beaucoup de façons, des codes à enchaînement qui suivent contours de pixel aux formats cannelés de vecteur.

V.2.1. Description locale

Les caractéristiques présentées précédemment sont calculées de façon globale sur toute l'image. Cependant, les utilisateurs ont besoin encore de chercher un objet ou une partie dans la base d'images. Et les caractéristiques globales ne peuvent pas donner les résultats désirés. Supposant qu'une image composée de plusieurs objets, avec les caractéristiques (couleurs, textures et formes) le vecteur de caractéristiques globales extrait à partir de l'image entière n'a pas assez des informations locales (les objets). Donc par conséquent, le système doit utiliser des caractéristiques locales. Il y a trois approches [Da98] :

Dans la première approche, on divise une image en utilisant une grille et les caractéristiques de chaque bloc sont calculées pour chaque case de cette grille. Par exemple l'image est divisée en 8 zones de même taille et les caractéristiques sont calculées pour chacune de ces zones.

Dans la deuxième approche, on fait la segmentation l'image pour la diviser en les régions locales. On souhaite que chaque région locale soit plus proche d'un objet dans l'image. Et puis on calcule les caractéristiques pour chacune des régions extraites. C'est difficile à choisir une "bonne" méthode de segmentation valide pour toutes les images de la base d'images.

Dans la troisième approche, on s'intéresse les points d'intérêts c'est à dire en se basant sur la Description locale d'images à base des points d'intérêts

V.3.1. Description spécifiques

Les signatures citées précédemment donnent une information générale pour une image. Pour des applications où les images présentent moins de variance de l'une à l'autre, ces descripteurs ne sont pas suffisants, notamment dans des applications comme la reconnaissance de visages et la reconnaissance d'empreintes digitales, Des descripteurs spécifiques ont été proposés pour ces domaines particuliers en utilisant les connaissances expertes.

Les caractéristiques de visages comme les caractéristiques de la couleur de peau, les positions des yeux, de la bouche,...sont extraits. Ensuite, le système fait la comparaison avec les échantillon.

VI. Problèmes liés à l'indexation

Les contraintes principales imposées par le problème de l'indexation d'images par le contenu sont de trois types :

1. La première contrainte concerne le temps de calcul, la réponse à une requête doit être rapide.
2. La deuxième contrainte est très fortement liée à la précédente et concerne la taille de la mémoire du système. Le problème concerne la mémoire primaire (mémoire RAM) requise pour le chargement des index des images au moment de la recherche et la mémoire secondaire (disque dur) pour le stockage des index de la base. Avec la baisse du coût de ces mémoires, cette dernière contrainte peut apparaître comme mineure, cependant la taille des index est une composante importante particulièrement dans le cas des grandes bases d'images (composées de plusieurs millions d'images).
3. La troisième contrainte est liée à la satisfaction du but de l'utilisateur puisque un système qui ne répond pas aux attentes de ce dernier n'est d'aucune utilité. Deux images similaires pour l'utilisateur doivent posséder des index proches.

VII. Similarité dans le domaine des images

Il existe plusieurs manières de rechercher, dans une base d'images, une image similaire à une image requête.

Signature d'images

La signature d'une image est une caractéristique visuelle qui peut être la valeur des pixels, ainsi plus les valeurs des pixels homologues de deux images sont proches, plus les deux images se ressemblent. Cette signature est représentée d'une manière générale par un vecteur descripteur de dimension N composants réels selon la technique d'indexation d'image utilisé.

C'est pourquoi ils ont définie plusieurs mesures de similarité entre deux images I_1 et I_2 représentés respectivement par deux vecteurs descripteur V_1 et V_2 sous forme d'une fonction $D(V_1, V_2)$ pour plusieurs type de distance suivantes :

Distance euclidienne

Distance entre deux vecteurs V_1, V_2 :

$$d(V_1, V_2) = \sqrt{\sum_{i=1}^N (V_{1,i} - V_{2,i})^2}$$

N la taille de vecteur.

Distance de Manhattan

Distance entre deux vecteurs V_1, V_2 :

$$d(V_1, V_2) = \sum_{i=1}^N (|V_{1,i} - V_{2,i}|)$$

N la taille de vecteur.

VIII. Systèmes de recherche d'images

VIII.1. Introduction

Un système robuste de caractérisation et de recherche d'images est l'un des outils essentiels pour gérer les nouveaux systèmes d'information multimédia. L'utilisateur doit être capable de trouver les images souhaitées dans une base de données d'images.

La plupart des systèmes de recherches d'images développés utilisent des mots clés ou des descriptions textuelles pour caractériser chaque image de la base de données d'images. Ce type de caractérisation comporte plusieurs inconvénients. En effet, la description textuelles est une tâche difficile à élaborer et ne peut pas présenter fidèlement toutes les caractéristiques des images. Pour dépasser ces problèmes et pour améliorer la recherche des images, les chercheurs ont focalisé leurs travaux sur la caractérisation par les attributs visuels des images au lieu des descriptions textuelles.

VIII.2. Principe

Une première classification des méthodes de recherche d'images peut être proposée en considérant le but visé par l'utilisateur. Cette classification distingue trois grandes catégories :

- La recherche associative (search by association).
- La recherche cible (target search).
- La recherche de catégorie (category search).

En premier lieu, la « recherche associative » est un type de recherche de type vague procédant par exploration de la base d'images. L'utilisateur ne possède pas dans ce cas d'exemple précis de ce qu'il recherche (il peut d'ailleurs ne pas savoir ce qu'il recherche), c'est l'exploration qui permet, au moyen d'une interaction système-utilisateur forte, de définir et d'affiner le but de la recherche.

La seconde approche, au contraire de l'approche précédente, la « recherche de cible » a un but clairement défini, à savoir retrouver une image particulière, précise que l'utilisateur a en tête, par exemple la photographie d'un objet donné.

Cette approche fait appel à un processus interactif permettant de converger vers l'image cherchée.

La troisième approche, la « recherche de catégorie » a pour objectif de retourner une, plusieurs ou bien toutes les images appartenant à une catégorie donnée. C'est l'utilisateur qui fixe les limites des catégories en interagissant avec le système.

Une seconde classification des techniques de recherche d'images permet de distinguer deux grandes méthodes :

- La recherche par similarité.
- La modélisation de la pertinence.

La recherche par similarité ou la recherche par l'exemple est toute recherche initiée par un exemple fourni par l'utilisateur au système. Concrètement, La requête peut être une « image-type », une région dans une image-type, un croquis, une requête hybride image-texte ou autre. La recherche se base sur une mesure

de similarité ou de dé similarité calculée entre les images cibles et la requête. En conséquence, les images résultats retournées par le système sont « proches » de l'exemple, autrement dit, similaires à la requête au sens de la mesure adoptée. Cette approche est donc fortement dépendante de la requête, qui peut être vue comme un « point d'ancrage » (dans l'espace de recherche) pour le calcul de similarité, basé sur une fonction propre au système.

La recherche par modélisation de la pertinence n'est généralement pas initiée par un exemple. Une valeur traduisant l'adéquation à la requête, généralement une probabilité, est affectée aux images. La densité de la probabilité construite pour l'ensemble de la base est initialisée uniformément, puis mise à jour sur le principe du bouclage de pertinence. Les modes qui émergent de la densité de probabilité, au fil des itérations, correspondant théoriquement aux images recherchées.

Donc un système de recherche par similarité d'images nécessite trois composantes :

1. Extraction des attributs significatifs de chaque image de la base et de l'image requête et représentation efficace de ces attributs en machine. A chaque image (ou classes d'images) correspond donc un ensemble d'attributs appelé index ou signature d'image.

2. Définition d'une ou plusieurs métriques de similarité perceptuelle ment significatives et rapidement calculables entre deux signatures d'images.

C'est au sens de cette métrique que le système doit retrouver rapidement les images les plus similaires à l'image requête.

3. Elaboration d'une interface homme-machine ergonomique pour les requêtes, la visualisation et la navigation. Au besoin, l'interface permet également à l'utilisateur de raffiner sa requête pour obtenir l'image désirée.

VIII.3. Exemple de système de recherche d'images (SRI)

VIII.3.1. Le système QBIC

IBM a présenté à l'intention des professionnels un système de gestion de document (textes, images, son) baptisé Content Manager. L'une de ses

particularités est Qbic, un moteur de recherche graphique qui permet de retrouver une image en utilisant l'analyse des couleurs, des contours ou des textures.

Développé depuis une dizaine d'années dans les laboratoires d'Almaden Etats-Unis, Qbic signifie « query by image content », qu'on peut traduire par « recherche par contenu graphique ».

A partir d'une bibliothèque de photos, le logiciel crée un catalogue d'indexation qui rassemble, pour chaque image, les informations mathématiques sur les textures (rayé, uni), les pourcentages de bleu, de vert et de rouge, le positionnement des formes ou des textures. « Le musée de l'Hermitage de St Peters bourg a retenu Qbic pour indexer environ 3000 œuvres en haute définition sur son site ».

L'internaute peut donc, en dessinant la forme de son choix et en jouant sur les couleurs, tenter de retrouver un tableau de maître. L'interface de site de musée permet de dessiner des cercles ou des rectangles. A l'essai, la représentation d'un ciel, avec un cercle jaunâtre en haut et différentes parties ovales ou rectangulaires en bleu et vert, le haut de l'écran renvoient effectivement vers des paysages, mais tous les coups ne sont pas gagnants.

Avec la méthode des couleurs, un bloc inférieur noir associé à un bloc supérieur jaune revolé vers un panorama de XVII^{ème} siècle, aux tons très proches.

La présence d'autres tableaux graphiquement plus éloignés laisse tout de même perplexe.

C'est qu'il reste des progrès à faire. En France, de nombreux laboratoire développent des moteurs graphiques mais ils sont encore peu intuitifs, et les réponses sont souvent hors sujet ou classées bizarrement.

« Un spécialiste de la recherche arrive tout de même à de bons résultats », défend José Martinez, maître de conférence à l'école polytechnique de l'université de Nantes. Selon lui, il existe une quantité impressionnante de moteurs expérimentaux, sans compte les trois versions commerciales existantes qui sont Virage, Exalibur RetrievalWare et le fameux Qbic d'IBM. « Dans une base 1200 images, je peux retrouver l'une d'elles après une douzaine d'itérations » en jonglant avec les paramètres.

Pour le grand public, Qbic peut rechercher les photos de l'album de famille, par exemple en indiquant la forme d'un chapeau.

VIII.3.2. Le système IKONA

Le projet IMEDIA a donné naissance à un nouveau moteur (IKONA) d'indexation et de recherche par le contenu dans la base de données images, basé sur une architecture Client/serveur: IKONA/MAESTRO.

Ce logiciel peut déterminer les similitudes à travers les milliers d'images de sa base de données.

Afin de traiter ces bases d'images, IKONA inclut une technique de contrôle qui permet à l'utilisateur de raffiner sa requête.

Comme il peut donner la possibilité à l'utilisateur de choisir une partie d'une images et le système recherchera les images (ou les parties d'images) qui sont visuellement semblable à la partie choisie.

Cette interaction permette à l'utilisateur de préciser au système quelle partie ou objet particulier est intéressant dans l'image.

Les quatre éléments de l'environnement IKONA du projet IMEDIA sont les suivants :

- GLOBAL SIG (version 1.0): signatures globales pour la recherche approximative d'images par similarité visuelle (distribution spatiale de la couleur, structure et texture).
- HCP (version 1.0) : le logiciel permet d'effectuer des requêtes partielles sur les images avec une sélection interactive de la zone d'intérêt. Il génère une description locale de l'image fondée sur des points caractéristiques sans effectuer une segmentation préalable.
- MAESTRO (version 1.0) : Serveur de calcul des signatures et des similarités pour la recherche d'images par contenu visuel.

- IKONA-client (version 1.0) : Client interactif pour la recherche d'images par similarité visuelle avec le serveur MAESTRO.

Conclusion

Dans ce chapitre on a pris un coup d'œil sur le principe de la recherche des images par le contenu dans une base de données des images sur la base d'utilisation le principe d'indexation d'image et l'indexation de l'ensemble des images d'une base de données cela nous permet de trouver les images similaires à une image requête tout en manipulant ces descripteurs visuels extraits à partir de l'image initiale selon les techniques d'indexation afin de mesurer la distance entre deux vecteurs descripteurs associés à deux images .

CHAPITRE2

La structure de l'Arbre Quaternaire Générique

La structure de l'Arbre Quaternaire Générique

I. Introduction

Beaucoup d'applications, sont amenées à générer, stocker et gérer des images similaires. De nombreuses mesures de similarités (sur la couleur, la texture ou la forme, par exemple) ont été proposées. Ces images similaires apparaissent dans les systèmes de traitement d'images [6] où une nouvelle image correspond au résultat d'une opération ou d'une série particulière d'opérations sur une image initiale [7]. La conservation des images permet aux utilisateurs de ce type d'application d'effectuer des retours en arrière lorsque, par exemple, une opération a trop modifié l'image sur laquelle elle a été appliquée. Dans les Systèmes d'Information Géographique, des images similaires sont créées à partir d'images satellites ou d'images radar de la même région, prises à intervalle régulier, an de suivre l'évolution de la région. Dans le domaine de l'imagerie médicale, des images représentant les mêmes types de pathologie peuvent être considérées comme similaires.

Le stockage des images impose un grand espace mémoire. Une image peut, en effet, occuper de quelques kilo-octets à plusieurs giga-octets. De plus, une base de données d'images est supposée contenir un grand nombre d'images, permettre de les lire, et d'effectuer des opérations sur les images. Par conséquent, il est nécessaire de construire des outils qui stockent les images de manière à minimiser l'espace occupé, tout en permettant de gérer, lire, interroger, ou modifier ces images.

On distingue, dans la littérature, différentes approches pour la gestion des images, selon les fonctions qu'elles proposent. Certaines approches extraient de l'information ou des métadonnées sur les images, par exemple les caractéristiques visuelles telles que la couleur, la forme ou la texture. Ces approches permettent de faire de la recherche d'images par le contenu (en anglais content-based image retrieval).

Elles ne s'intéressent ni à l'optimisation du stockage des images ni à leurs modifications éventuelles. Les outils proposés par ce type d'approches, tels que QBIC, sont appelés systèmes d'information d'images, par opposition aux bases de données d'images, c'est-à-dire aux Systèmes de Gestion de Base de Données permettant de représenter et de gérer aussi bien des données alphanumériques que des images. En revanche, d'autres approches proposent des mécanismes de compactage des images, à l'aide d'ondelette ou par partage d'information entre les structures représentant ces images. Ces approches proposent principalement des solutions pour le stockage des images mais ne prennent pas en compte les opérations sur les images, à l'exception de la lecture d'une image ou d'une séquence d'images. A notre connaissance, il n'y a pas dans la littérature de propositions permettant à la fois de minimiser l'espace de stockage des images tout en laissant l'opportunité aux utilisateurs d'effectuer des traitements sur ces images [10].

Dans ce chapitre, une structure de données permettant de gérer les images est proposée. Cette structure, appelée Arbre Quaternaire Générique, stocke des images similaires organisées en arbre quaternaire. Un arbre quaternaire (ou quad-tree) est une structure de données efficace pour représenter les images à deux dimensions [8, 9]. L'Arbre Quaternaire Générique minimise l'espace de stockage, par partage de parties communes entre images. Il permet d'appliquer des opérations sur les images, comme la comparaison d'images.

II. L'Arbre Quaternaire Générique

II.1. Définition

Un arbre quaternaire est une structure de données permettant de représenter des images à deux dimensions. Les images sont découpées en quadrants ou carrés réguliers selon un critère de découpage donné (l'homogénéité de la couleur des pixels, la texture, etc.). Un arbre quaternaire peut être implanté de manière hiérarchique, à l'aide de pointeurs, ou de manière linéaire, sous forme d'une liste de valeurs. Certaines opérations de base de traitement d'image (ex. le complémentaire d'une image, l'union ou l'intersection de deux images) sont applicables sur des images organisées en arbre quaternaire. Plusieurs approches proposent d'optimiser le stockage d'images, en appliquant des mécanismes de chevauchement sur les arbres quaternaires représentant les images. Ces mécanismes peuvent s'appliquer quelle que soit la manière avec laquelle les arbres quaternaires ont été implantés. Ces approches sont principalement orientées vers la gestion et le compactage d'images de séquences vidéo [10].

II.2. Principes de l'Arbre Quaternaire Générique

II.2.1. Le concept de partage entre les arbres quaternaires

L'arbre quaternaire générique est basé sur le principe de partage de régions (quadrants) entre images. Soit I_m un ensemble des images. Si un quadrant q a la même valeur dans un ensemble $I'_m \subset I_m$ cette valeur n'est stockée qu'une seule fois dans la base et est associée à l'ensemble des identificateurs des images I'_m . Dans ce cas on parle de partage explicite, parce que l'identificateur de chaque image partageant cette valeur apparaît explicitement dans la liste des images associées à cette valeur. Si les images de l'ensemble I_m sont organisées en arborescence, chaque image excepté la racine de l'arbre, a une mère unique et un nombre indéfini d'images filles. Par conséquent la règle de partage implicite suivante peut être introduite : " excepté lorsque l'identificateur d'une image i est implicitement associé avec une autre valeur v , l'image i partage implicitement la valeur associée à son image mère. Si l'arbre organisant les images est stocké, cette règle de partage implicite permet une représentation compacte de l'ensemble des images, en particulier lorsqu'un grand nombre d'images partagent des valeurs de quadrants dans plusieurs branches de l'arbre [10].

II.2.2. Arbre Quaternaire d'Image

Définition

C'est une structure hiérarchique, dite aussi quad-tree, construite par division récursive de l'espace en quatre quadrants disjoints de même taille [11], en fonction d'un critère de découpage (exemple : homogénéité de la couleur) de telle sorte que chaque nœud de l'arbre quaternaire représente un quadrant dans l'image. L'image entière est le nœud racine, si une image n'est pas homogène par rapport au critère de découpage, le nœud racine de l'arbre quaternaire représentant l'image a quatre fils représentant les quatre premiers quadrants de l'image, un nœud d'arbre quaternaire est dit feuille, si le critère d'homogénéité est vérifié sinon le nœud est interne [12].

Critères de découpage

La première idée qui vient à l'esprit lorsque l'on veut découper une image est de la diviser en deux, puis en quatre, puis en huit et ainsi de suite. Ce découpage par pavages carrés réguliers est appelé quadtree. Ce terme désigne un arbre (au sens structure de données du terme) composé de zones carrées qui composent ses feuilles. Nous avons ici une représentation typiquement récursive : soit une image que l'on découpe en quatre et pour chacun des morceaux on réapplique ce découpage. Ce processus peut être itéré jusqu'à la notion de pixel (qui prend ici tout son sens). Si l'image fait une taille $N \times N$, alors le nombre de découpage (le nombre de niveaux dans l'arbre) est de $N+1$ [13].

L'application de ce découpage peut se faire sur une image binaire ou en niveaux de gris. Chaque bloc représente alors une zone dite "homogène". Le critère d'arrêt du découpage est simple sur une image binaire : si la zone contient à la fois des pixels noirs et des pixels blancs, alors, il faut la redécouper de nouveau. Sinon, il faut s'arrêter et considérer que la zone est homogène [13].

Dans une image en niveaux de gris et, pire encore, sur une image couleur, les critères d'arrêts sont loin d'être évidents. Différentes approches ont été mises en œuvre et nous pouvons citer, entre autres, l'analyse des textures de la scène, les calculs de variances et de moyennes, les approches de la théorie de l'information ainsi que d'autres critères plus "physiques" comme le critère d'hystérésis. La construction d'un quadtree est donc liée à une recherche d'homogénéité (variance minimale, attributs semblables, etc.) [13].

La représentation d'une image en arbre quaternaire

L'image est découpée récursivement en quatre quadrants ou carrés réguliers (Figure 2.1), tels que chaque quadrant soit représenté par un nœud de l'arbre quaternaire. La racine représente l'image dans sa totalité. Elle a quatre nœuds fils représentant les quatre principaux quadrants (00,01,02,03) (Figure 2.2). Chaque nœuds fils possède à son tour quatre fils, représentant ainsi les seize sous quadrants de l'image initiale, puis, ces quadrants sont à leur tour découpés en quatre, et ainsi de suite. Les nœuds feuilles de l'arbre quaternaire ne sont pas tous au même niveau. Ils représentent les régions de l'image. Il existe plusieurs fonctions permettant de savoir à quelle partie de l'image correspond un nœud de l'arbre quaternaire, et inversement. Ces fonctions utilisent une identification particulière des nœuds. La racine de l'arbre quaternaire, représentant l'image tout entière, est identifiée, par exemple par 0. Ses fils, représentant les quatre premiers quadrants de l'image, ont pour identificateurs respectifs 00; 01; 02 et 03. Chaque fils du nœud 00 a pour identificateur $00x$, $x \in [0; 3]$, chaque fils du nœud 01 a pour identificateur $01x$, $x \in [0; 3]$, etc. Cette numérotation est employée dans la (Figure 2.1). Il s'agit d'une numérotation en $Z [10]$.

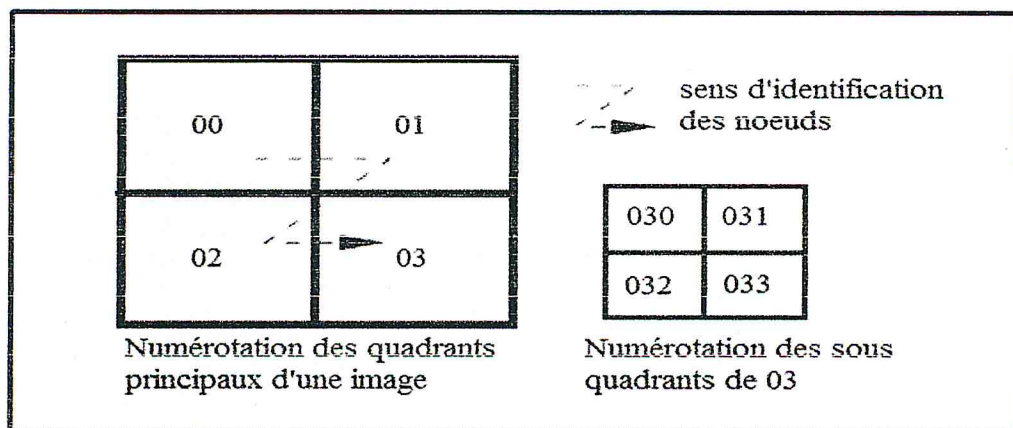


Figure 2.1 - Numérotation des quadrants

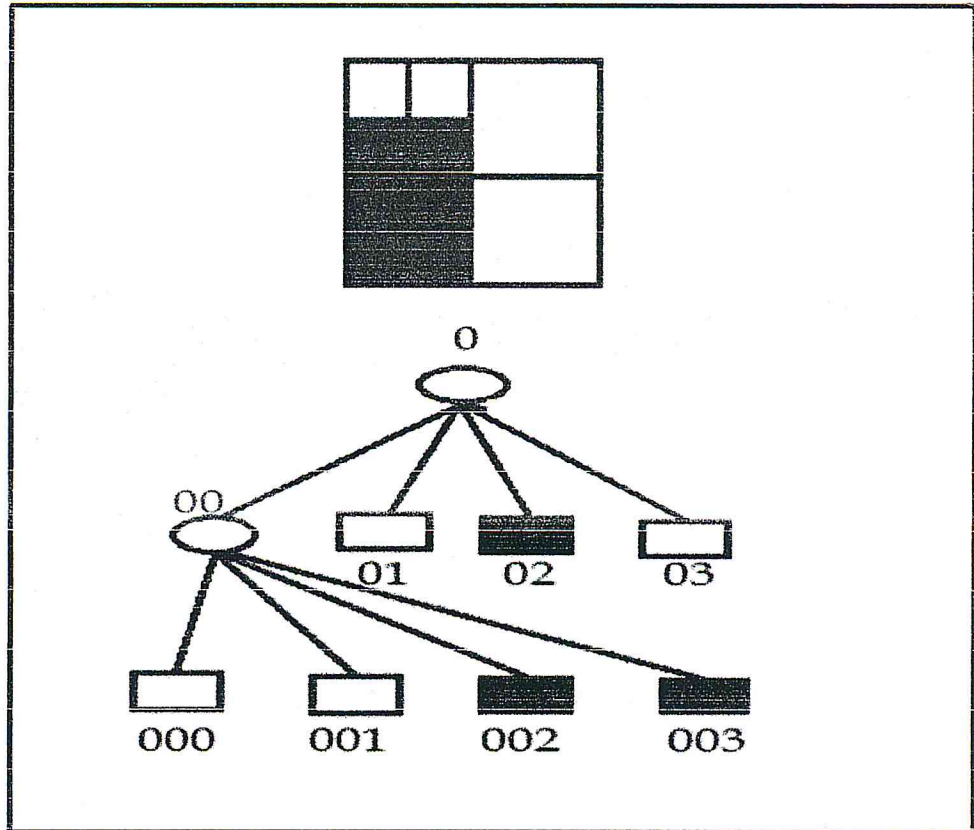


Figure 2.2 - représentation d'une image en arbre quaternaire

Similarité entre images

Les images sont regroupées, dans la base de données ; en fonction d'une distance de similarité entre les arbres quaternaires qui représentent. Cette distance, appelée *Q-similarité*, est proposée afin d'optimiser le stockage des images dans la base. La distance de Q-similarité entre deux images est définie par le nombre des nœuds différents (de même identificateur et de valeur différente) dans les deux arbres quaternaires représentant les images, divisé par le nombre d'identificateurs de nœud (sans doublon) apparaissant dans l'union des nœuds des arbres quaternaires des images [10].

De manière plus formelle, on note $S(i, i')$ l'ensemble des nœuds différents entre les arbres quaternaires des images i, i' . On note $U(i, i')$ l'ensemble (sans doublon) des identificateurs de nœud apparaissant dans les arbres quaternaires des images i et i' . on note $| S(i, i') |$ (resp. $| U(i, i') |$) le nombre d'éléments de $S(i, i')$ (resp. $U(i, i')$) [10].

La distance de Q-similarité des images i et i' notée $d(i, i')$, est calculée par l'équation suivante :

$$D(i, i') = \frac{|S(i, i')|}{|U(i, i')|} = \frac{\text{nombre de nœuds différents}}{\text{Total des identificateurs de nœud (sans doublon)}}$$

- $d(i, i') \in [0, 1]$, c'est une distance normalisée, c'est-à-dire :
 $d(i, i) = 0$, $d(i, i') = d(i', i)$ et $d(i, i') \leq d(i, j) + d(j, i')$ (inégalité triangulaire).
- $d(i, i') = 0 \Leftrightarrow$ les arbres quaternaires des images i et i' partagent tous leurs nœuds.
- $d(i, i') = 1 \Leftrightarrow$ les arbres quaternaires des images i et i' ne partagent aucun nœud.

Exemple 2.1 [10] : La distance de Q-similarité des images **a** et **b**, dont les arbres quaternaires sont représentés sur la (Figure 2.3), est égale à :

$$d(a, b) = \frac{|S(a, b)|}{|U(a, b)|} = \frac{5}{9}$$

- $|U(a, b)| = 9$, car les nœuds 0, 0x et 03x, $x \in [0; 3]$ apparaissent dans l'union des identificateurs des nœuds des arbres quaternaires des images **a** et **b** (voir Figure 2.3).
- $|S(a, b)| = 5$, car les nœuds 03 et 03x, $x \in [0; 3]$ sont différents dans les deux arbres quaternaires.

Les images **a** et **b** sont 5/9-Q-similaires.

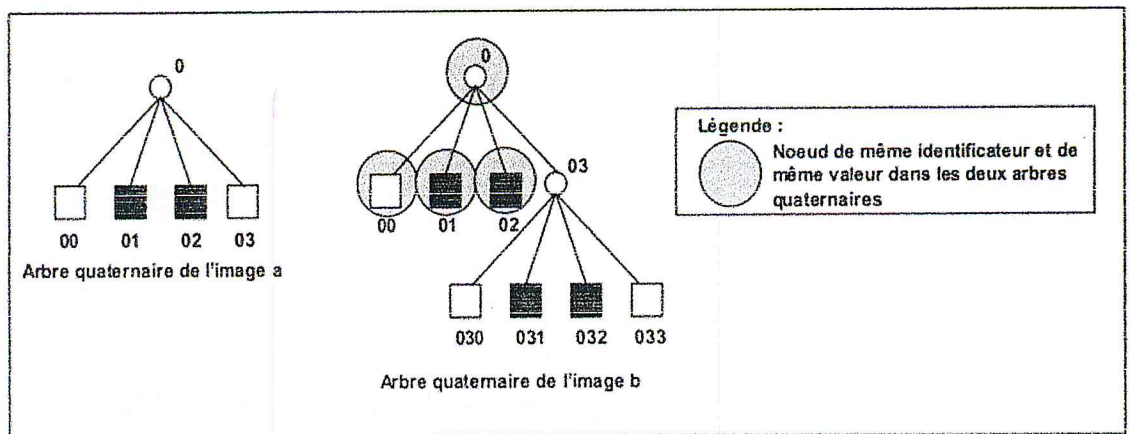


Figure 2.3 – le Calcul du coefficient de Q-similarité des images a et b

II.2.4. Arbre Quaternaire Générique SPI

L'Arbre Quaternaire Générique SPI (ou Arbre Quaternaire Sans Partage Implicite) est un Arbre Quaternaire Générique construit sans tenir compte du partage implicite éventuel entre les images, c'est-à-dire en considérant les images indépendamment les unes des autres. Cette structure est construite à partir des arbres quaternaires représentant les images. Chaque image de l'ensemble est représentée par un arbre quaternaire. Tous les nœuds des arbres quaternaires sont identifiés de la même manière, de sorte que l'on puisse faire la correspondance entre les nœuds des différents arbres quaternaires. L'Arbre Quaternaire Générique SPI est construit avec autant de nœuds génériques qu'il y a de nœuds d'identificateurs différents dans l'ensemble des arbres quaternaires. Un nœud générique n a autant de lignes qu'un nœud n a de valeurs dans les arbres quaternaires. A chaque valeur v d'un nœud générique n est associée la liste des identificateurs d'image pour lesquelles l'arbre quaternaire correspondant contient un nœud identifié par n et de valeur v [10].

Exemple 2.2:

La Figure 2.3 représente quatre images organisées en arbre quaternaire. L'Arbre Quaternaire Générique SPI stockant les arbres quaternaires de ces images est représenté par la Figure 2.4.

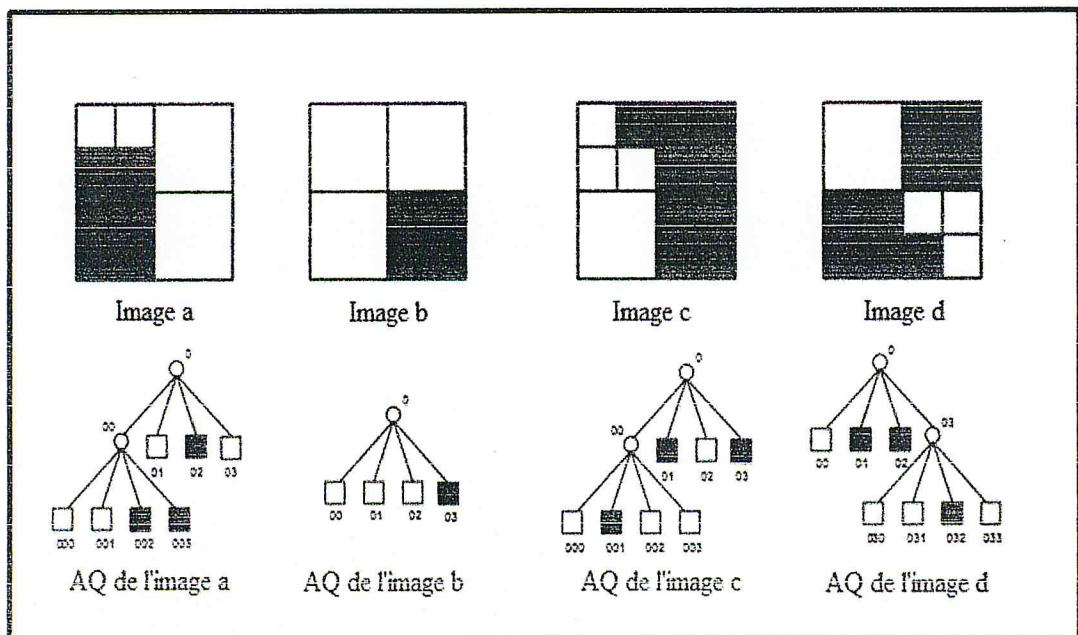


Figure 2.4 - La représentation de quatre images en arbre quaternaire [10].

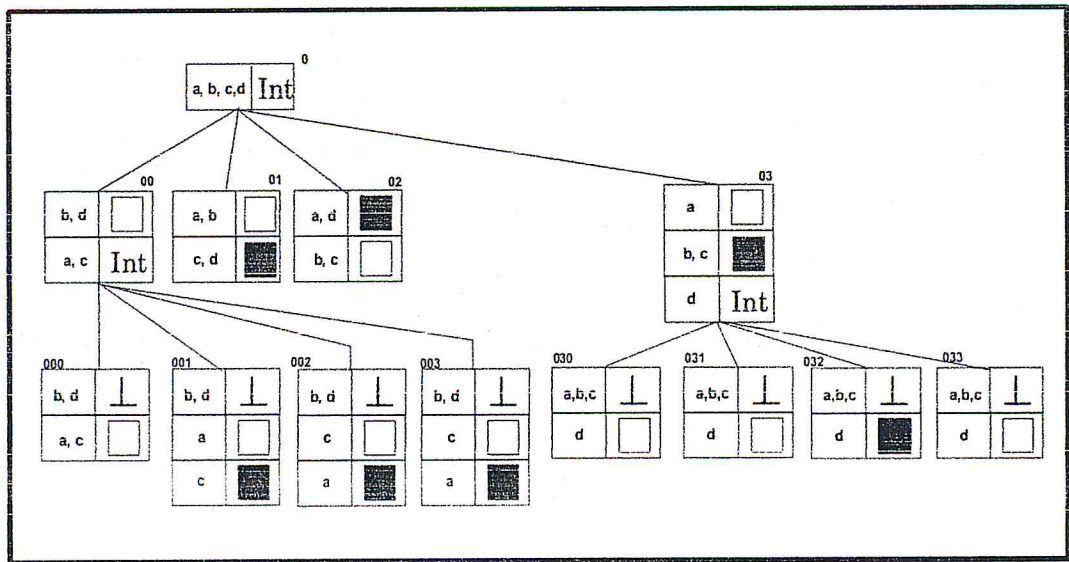


Figure 2.5 -L'Arbre Quaternaire Générique SPI des images de la Figure 2.4 [10].

II.2.5. Nœuds génériques

La représentation et le stockage d'un ensemble des images similaires sont effectués dans un arbre quaternaire générique, dont les nœuds sont appelés nœuds génériques. Pour chaque nœud apparaissant dans l'arbre quaternaire d'une image, il existe un nœud générique ayant le même identificateur dans l'arbre quaternaire générique. Un nœud générique n représente tous les nœuds n des arbres quaternaires des images de la base. Il contient toute l'information nécessaire pour recomposer la valeur du nœud de même identification dans chaque arbre quaternaire [10].

La valeur d'un nœud est soit \perp qui signifie que le nœud n'existe pas, soit *Int* qui signifie qu'il est interne (il y a quatre fils).

Chaque nœud générique peut être vu comme un tableau ayant plusieurs lignes. Chaque ligne l de nœud générique n contient une liste d'identificateurs d'images et une valeur v de nœud d'arbre quaternaire. La valeur v signifie que tous les nœuds identifiés par n ont pour valeur v dans les arbres quaternaires des images dont l'identificateur i apparaît dans la liste. De plus par application de la règle de partage implicite, on déduit que les nœuds n des arbres quaternaires de toutes les images descendantes des images de la liste, dans l'arbre d'images partagent implicitement cette valeur, excepté lorsqu'un identificateur d'image descendante apparaît explicitement dans une autre ligne du nœud générique [10].

II.2.6. L'arbre d'images

En conséquence de la règle de partage implicite, les images représentées par un arbre quaternaire générique sont organisées à l'aide d'une structure arborescente particulière, l'arbre d'images. Lorsqu'une nouvelle image est insérée dans l'arbre d'images, elle est insérée comme fille de l'image dont elle est la plus similaire, c'est-à-dire dont la distance entre l'arbre quaternaire associée et celui de l'image à insérer est la plus proche de 0 [10].

II.2.7. Graphe de partage et déduction de l'Arbre d'Images

Principes du graphe de partage

Un graphe complet non orienté $G = (I_m, A)$ est construit à partir de l'Arbre Quaternaire Générique SPI. Ce graphe, nommé graphe de partage, est composé d'un ensemble I_m d'identificateurs d'image et d'un ensemble A d'arêtes. Chaque arête $\{i, j\}$ est étiquetée par le nombre de nœuds communs (c'est-à-dire de même identificateur et de même valeur) dans les arbres quaternaires représentant les images i et j [10].

Exemple 2.3: Le graphe G , déduit de l'Arbre Quaternaire Générique SPI de la Figure 2.6, est représenté dans la Figure 2.7.

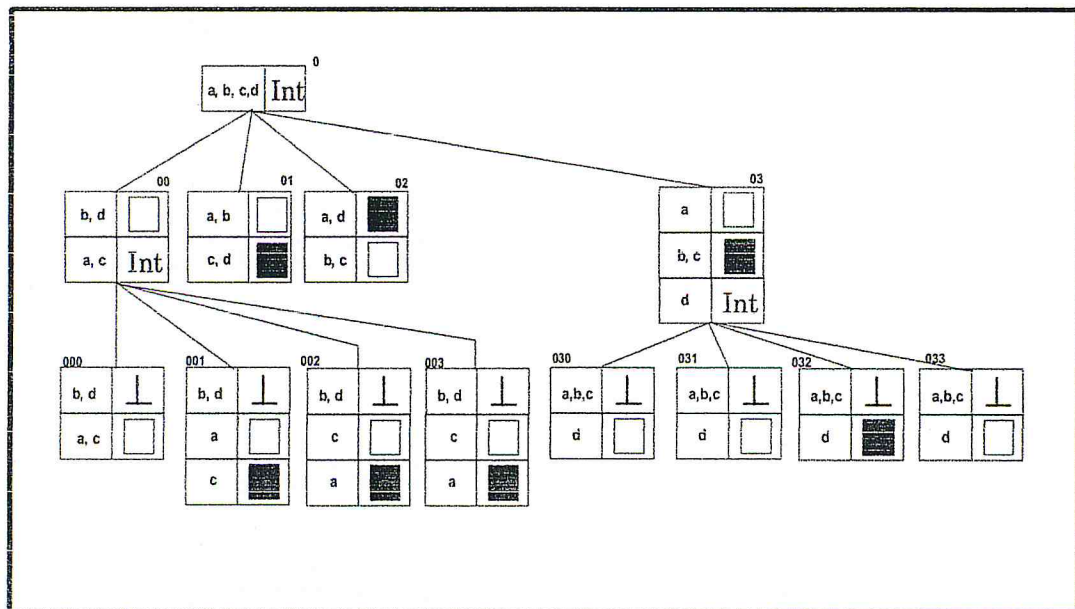


Figure 2.6 - L'Arbre Quaternaire Générique SPI des images de la Figure 2.4 [10].

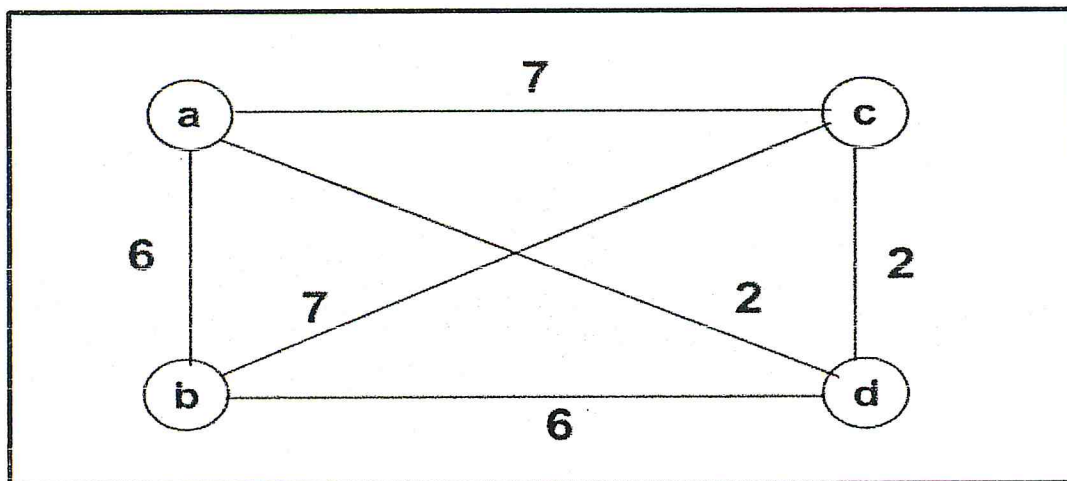


Figure 2.7 - Le graphe de partage de L'Arbre SPI de la Figure 2.6 [10].

Arc (a, b)=6, dans la Figure 2.8 on remarque qu'il y a 6 nœuds qu'il est pour valeur de a et b la même (numérotation bleu), même chose pour Arc (a, c) (numérotation rouge).

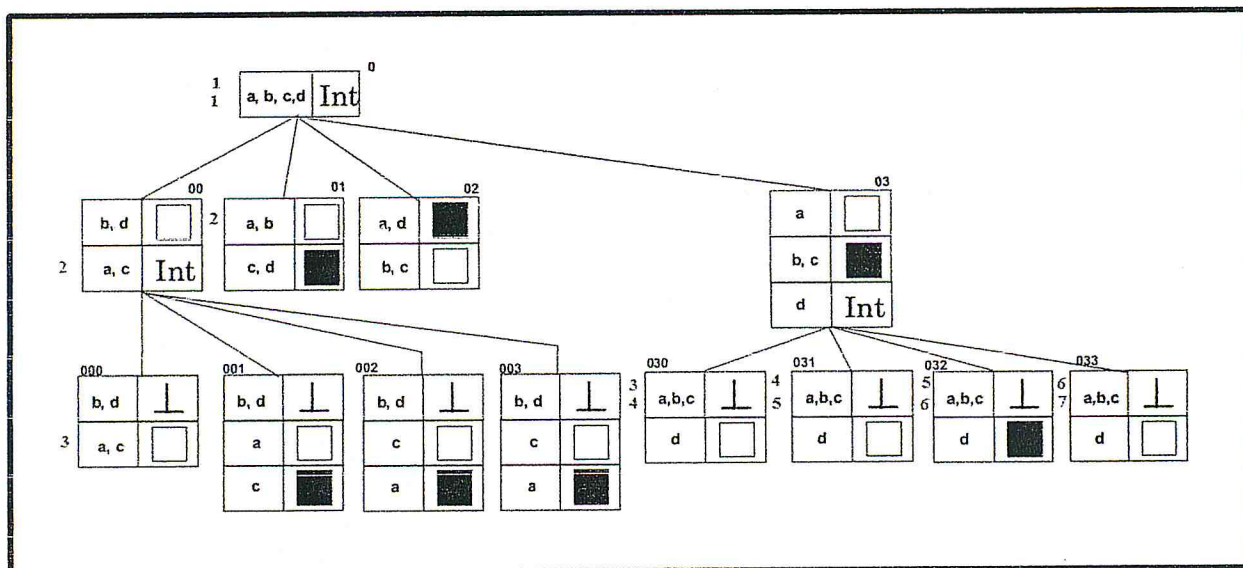


Figure 2.8 – Arc (a, b) dans l'arbre SPI

Recherche d'un arbre couvrant maximum

La recherche d'un arbre couvrant maximum du graphe G permet de connaître, pour chaque image i , avec quelle image de la base i partage le plus de valeur de nœud Un ou plusieurs arbres couvrants maximums peuvent être déduits du graphe G, en utilisant par exemple l'algorithme de Kruskal [Kru56]. L'algorithme de

Kruskal consiste à ordonner les arêtes par ordre croissant des étiquettes et à éliminer les arêtes du graphe de plus petite étiquette tant que le graphe reste connexe 3.

Algorithme de Kruskal

Principe :

On construit un sous-graphe en ajoutant des arêtes une par une. A chaque étape, on cherche l'arête de plus grand évaluation parmi celles que l'on n'a pas déjà explorées. Si elle ne crée pas de cycle, on l'ajoute au sous-graphe, sinon on la laisse de côté. On termine dès que l'on a sélectionné $n - 1$ arêtes ou qu'il ne reste plus.

Algorithme de Kruskal (arbre couvrant de poids maximal)

Kruskal(G)

- 1: $A =$ ensemble des arêtes du graphe G
- 2: $F = \emptyset$ (ensemble des arêtes de l'arbre couvrant)
- 3: Trier l'ensemble A des arêtes de G par évaluations décroissantes
- 4: pour $a \in A$ faire
- 5: si $F \cup \{a\}$ est acyclique alors
- 6: $F = F \cup \{a\}$
- 7: finsi
- 8: fin pour

Exemple 2.4: Nous allons appliquer l'algorithme de Kruskal au graphe de la(Figure2.9).

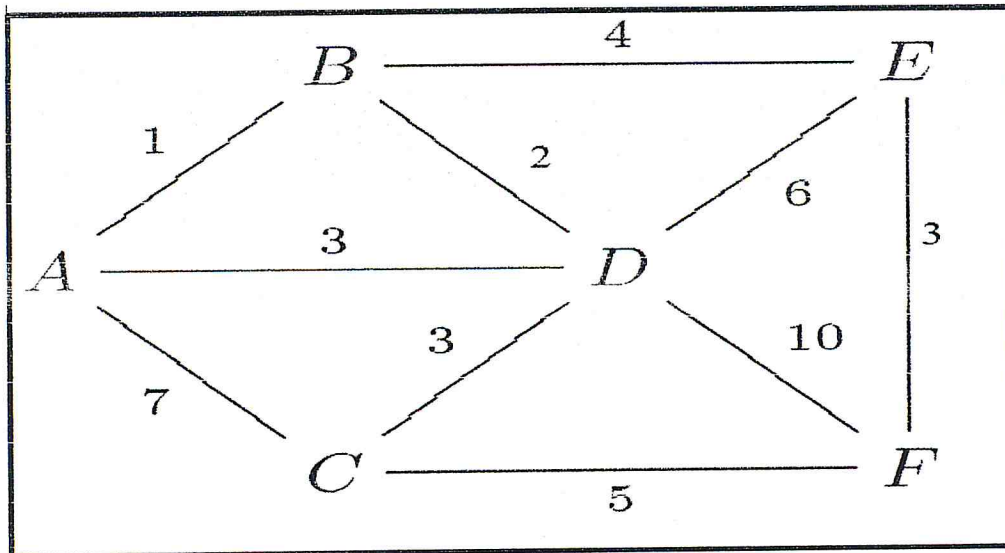


Figure 2.9 – Graphe de partage.

La première étape consiste à trier les arêtes par évaluations décroissantes. On obtient la liste ordonnée suivante :

Tableau 2.1 Liste ordonnée

(D,F)	(A,C)	(D,E)	(C,F)	(B,E)	(A,D)	(C,D)	(E,F)	(B,D)	(A,B)
10	7	6	5	4	3	3	3	2	1

On construit l'arbre couvrant en prenant les arêtes les unes après les autres dans l'ordre et en les ajoutant à l'arbre si elles ne créent pas de cycle.

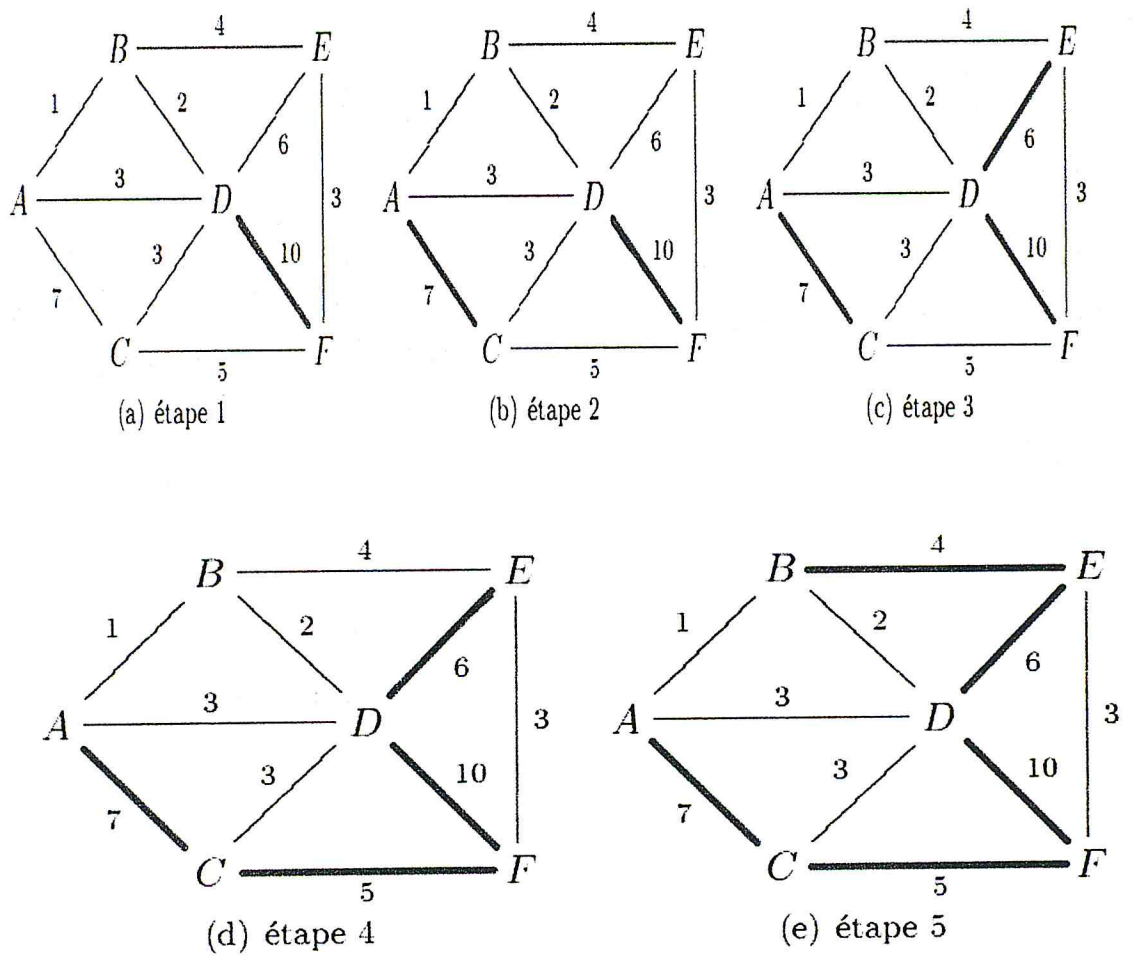


Figure 2.10 - les différentes étapes pour construire l'arbre couvrant maximal

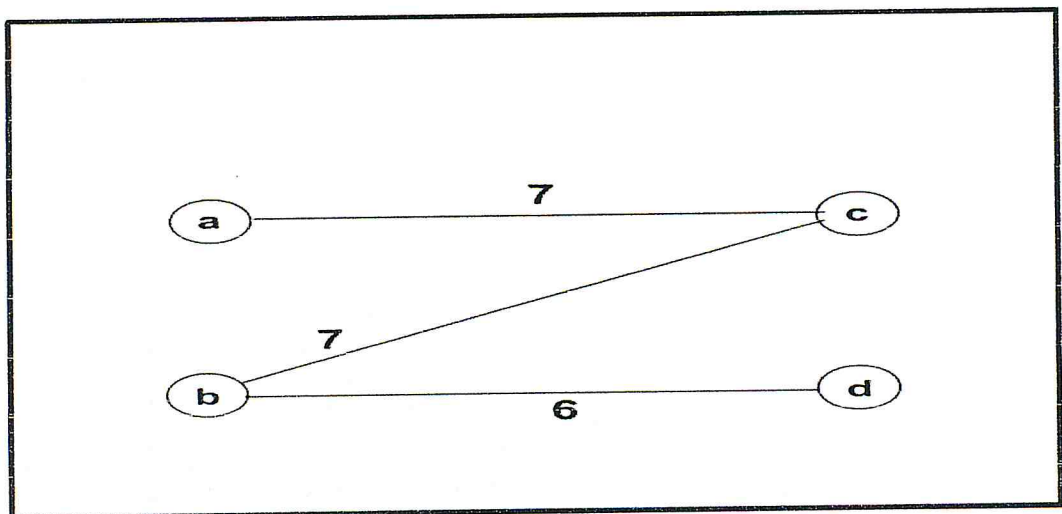


Figure 2.11 : l'arbre couvrant maximum pour le graphe dans la Figure 2.7 [10].

Déduction de l'Arbre d'Images

Une fois l'arbre couvrant choisi, on cherche un point de référence, c'est-à-dire une racine, an d'en déduire un ou plusieurs Arbres d'Images possibles pour organiser l'ensemble d'images (Figure 2.12). Un point de référence est un point qui minimise la longueur des chemins dans l'arbre. Les approches de [14] permettent de déterminer un point de référence dans un arbre sans racine. L'arbre ainsi obtenu correspond à l'Arbre d'Images recherché pour organiser les images de la base [10].

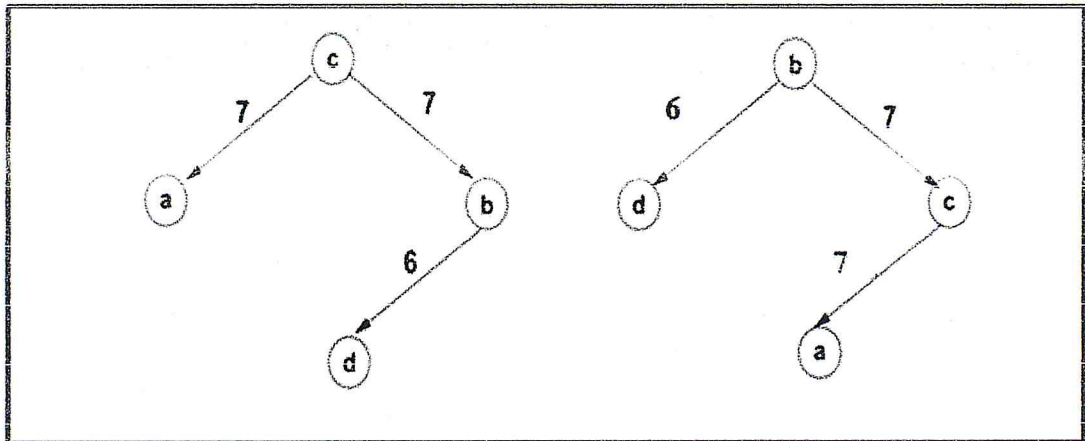


Figure 2.12 différents Arbres d'Images possibles [10].

II.2.8. Construction de l'Arbre Quaternaire Générique final

Consiste à supprimer tous les identificateurs d'image apparaissant simultanément, dans un même nœud générique, avec l'identificateur d'image mère dans l'arbre d'images. Par conséquent, pour chaque nœud générique n , si une image i apparaît dans une même ligne de nœud générique que l'image j , i étant fille de j dans l'arbre d'images, alors il faut supprimer i de la ligne. Il est à noter que cette suppression doit se faire en lisant l'arbre d'images des feuilles vers la racine en largeur d'abord, c'est-à-dire des images filles vers leur mère, pour transformer au fur et mesure le partage explicite entre mère et filles en partage implicite [10].

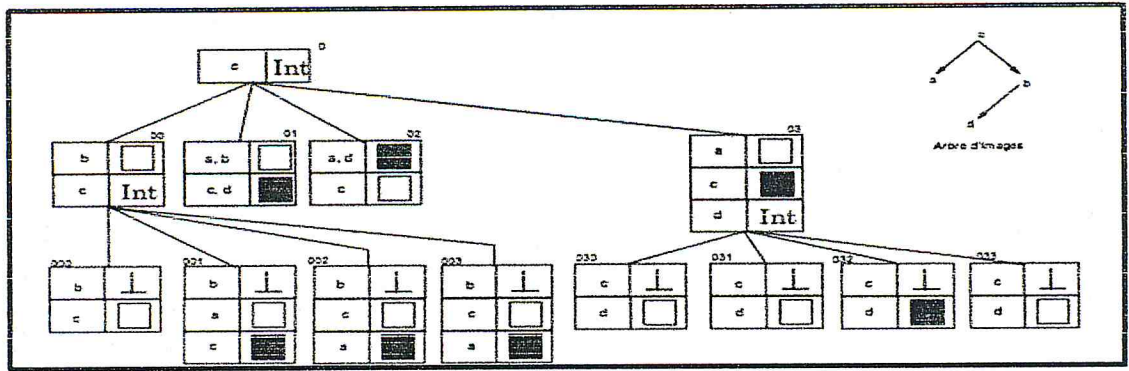


Figure 2.13 : L'arbre quaternaire générique final [10].

Conclusion

L'Arbre Quaternaire Générique présenté dans ce chapitre, permet d'optimiser le stockage d'images similaires organisées en arbre quaternaire, en évitant de stocker plusieurs fois des morceaux communs à plusieurs images. Cette optimisation, réalisée par le mécanisme de partage explicite, est d'autant plus considérable que les morceaux partagés ont une grande taille (dans le cas d'images en niveau de gris par exemple).

Le partage implicite permet, quand à lui, de diminuer le nombre d'identificateurs d'image apparaissant dans les nœuds génériques. Il diminue donc la taille des nœuds génériques.

CHAPITRE 3
Conception

Conception

I. Introduction

Dans ce chapitre on va expliquer toutes les méthodes et les techniques qu'on a utilisées dans notre travail pour construire l'arbre quaternaire générique.

II. Les images manipulées

On a séparé le travail entre les images en noir et blanc et les images en couleurs.

La dimension de l'image lue, carré et de dimension 2^N .

III. Construction de l'arbre d'images

III.1. Les images en noir et blanc

La Figure 3.1 représente les différentes étapes qu'on a faites pour construire l'arbre quaternaire générique pour les images noir et blanc.

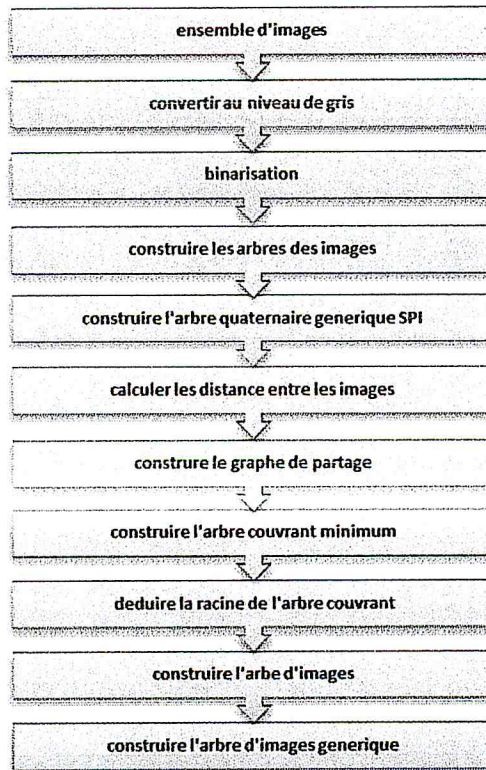


Figure 3.1 –processus de construction de l’arbre générique pour les images en noir et Blanc.

Conversion en niveau de gris

Un pixel dispose généralement des trois composantes RGB (en anglais : *Red*, *Green*, *Blue* ; en français : Rouge, Vert, Bleu). Un pixel gris a ses trois valeurs RGB identiques.

Dans notre travail on a utilisé une méthode simple pour convertir une image couleur en niveau de gris. Il suffit de calculer la moyenne des trois composantes RGB et d’utiliser cette valeur moyenne obtenue.

$$Gris = \frac{(Rouge + Vert + Bleu)}{3}$$

La Binarisation

La binarisation d'une image peut se faire à l'aide d'un seuil : les pixels dont le niveau de gris est en dessous du seuil deviennent noirs, et ceux au-dessus deviennent blancs.

Dans notre travail on a donné à l'utilisateur le choix d'un seuil de binarisation.

Construction des arbres d'images

Critères de découpage

Domination de couleur

On calcule le pourcentage des pixels noirs et blancs ensuite on définit un seuil.

Si l'un des deux pourcentages est supérieur ou égale au seuil on arrête le découpage et on associe la valeur du pourcentage le plus élevé à la région.

Exemple 3.1 :

Dans la région de Figure 3.2 le pourcentage des pixels noirs égaient à 75% et le pourcentage des pixels blancs égale à 25%.

0	1	0	1
1	0	1	1
1	1	1	1
0	1	1	1

Figure 3.2 -région d'une image.

Maintenant si le seuil égal à 80% le découpage ne s'arrête pas. La région doit se découper encore, mais si le seuil égale à 70% le découpage se termine et la région prend la valeur (n) qui Signifier noire.

Point fort : ce critère est utile quand on s'intéresse à la quantité des couleurs dans la région.

Point faible : ce critère est moins utile quand on s'intéresse à l'emplacement des couleurs.

Exemple 3.2:

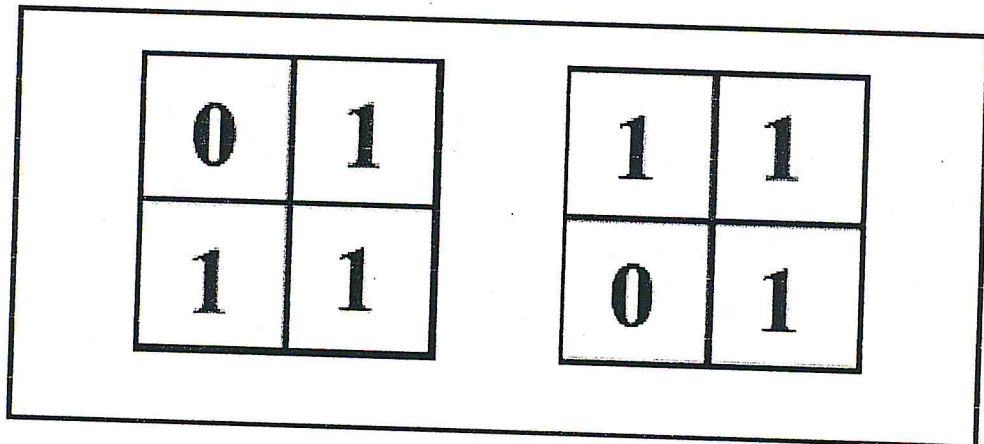


Figure 3.3 -Deux régions différentes ont le même pourcentage des couleurs.

Homogénéité

On a utilisé une méthode pour savoir si la région est homogène ou pas.

Explication de la méthode :

Pour savoir si la région de figure 3.4

est homogène on doit tester pour chacun des pixels dans la zone rouge ces huit voisins

0	1	0	1
1	0	1	1
1	1	1	1
0	1	1	1

Figure 3.4 - région d'une image

Si le voisin est égale à pixel courant on augmente le variable d'une unité ho sinon on augmente le variable d'une unité nho, à la fin on calcule le pourcentage d'homogénéité qui égale :

$$Pr = \frac{ho}{ho + nho};$$

Ensuite si (Pr) est supérieur ou égale à un pourcentage défini on arrête le découpage, sinon on découpe la région.

Pour la Figure 3.4 on aura :

0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1

H_o =nombre des cases bleus=21.

N_{ho} =nombre des cases rouges=11.

Pourcentage homogénéité= $21 / (21+11) = 65,625\%$

Point fort :

Ce critère est utile quand les couleurs de la région sont classifiés ou Structurés, c'est à dire les pixels noirs et les pixels blancs sont bien regroupés l'un a côté de l'autre. Ce critère est bien utilisé dans la segmentation pour découvrir les objets ou les formes dans la région.

Dans notre cas si la région est homogène (elle a des objets) donc on arrête le découpage si non on continue de découper jusqu'à trouver des régions contenant des objets.

Remarque : un objet c'est un regroupement des pixels de même valeur.

Valeur des nœuds de l'arbre d'image

Chaque nœud contient deux valeurs. Une pour le nom de la région et l'autre valeur « n » signifie (noire), « b » signifie blanc ou « Int » signifie qu'il est interne « il y a quatre fils ».

Niveau de découpage

Le niveau de découpage est un paramètre qui définit à quel niveau dans l'arbre d'image on doit s'arrêter.

Stockage de l'arbre image

On a proposé une méthode pour stocker l'arbre d'image dans un fichier d'extension (.aqn) de manière qu'on peut la récupérer.

Dans notre fichier on stocke juste les fils de l'arbre, cette technique permet d'éliminer tous les pères dans l'arbre ou pour bien précisé tous les nœuds qui ont une valeur « Int ».

Pourquoi on ne stocke pas les nœuds père ? Parce que si on a les fils on peut facilement déduire les parent, par exemple si on n'a la région 0001 en déduire qu'elle à le père 000 et grand-père 00.

Structure de fichier

Le fichier est un fichier texte composé de trois lignes :

- la Première ligne contient la dimension de l'image, elle est utile au moment de la récupération de l'image.
- la Deuxième ligne contient les régions feuille dans l'arbre qui ont la valeur « noir ».
- la Troisième ligne contient les régions feuille dans l'arbre qui ont la valeur « blanc ».

Exemple 3.3:

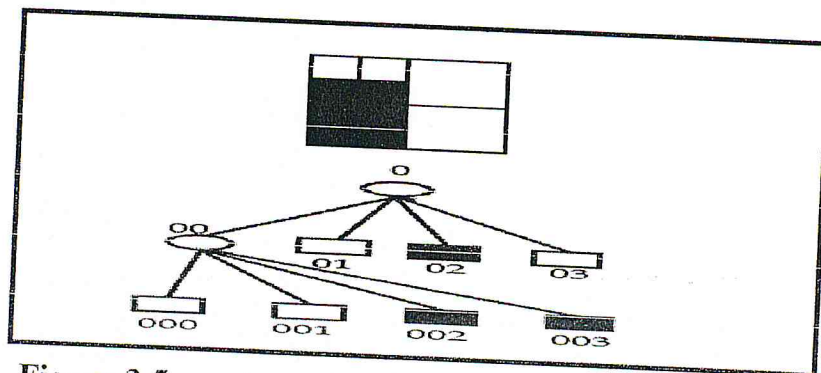
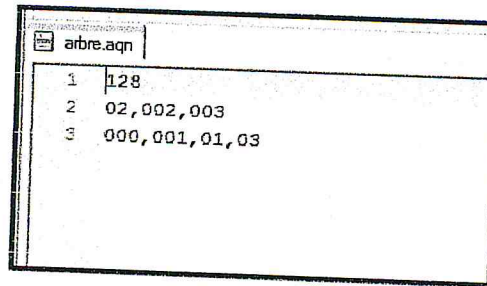


Figure 3.5 -représentation d'une image en arbre quaternaire

Fichier de stockage de l'arbre de Figure 3.5 :



```
arbre.aqn
1 128
2 02,002,003
3 000,001,01,03
```

Figure 3.6 –continue de fichier stockage.

Lecture de fichier de stockage

On a programmé une petite application qui permet d'afficher les fichiers de type (.aqn) « l'arbre d'image ».

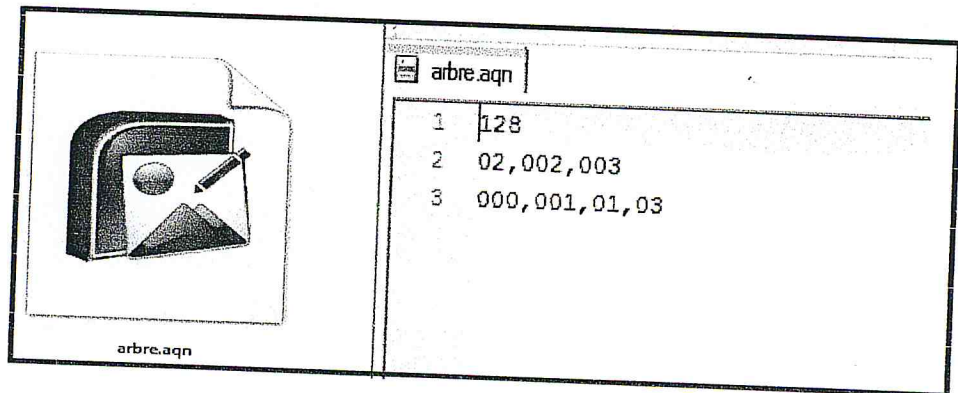


Figure 3.7 – fichier (.aqn)

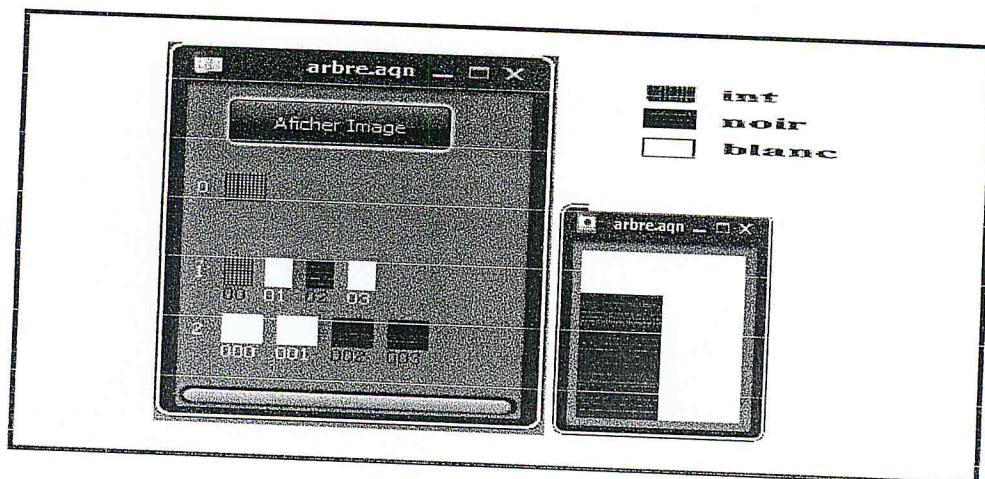


Figure 3.8 – Affichage de l'arbre et l'image.

Distance utilisé

On a utilisé la distance de Q-similarité qu'on a expliqué dans le chapitre 2

La distance de Q-similarité des images i et i' notée $d(i, i')$, est calculée par l'équation suivante :

$$D(i, i') = \frac{|S(i, i')|}{|U(i, i')|} = \frac{\text{nombre de nœuds différents}}{\text{Total des identificateurs de nœud (sans doublon)}}$$

Construction de l'arbre quaternaire générique spi

On utilise les arbres d'images pour construire l'arbre spi de la même manière qu'on a expliqué dans le chapitre 2.

Chaque nœud de l'arbre contient le nom de région et une valeur noire, une valeur blanche et une valeur Int associé avec des identificateurs des images.

On a proposé une représentation de l'arbre par une matrice comme ce si :

	n	b	Int
00			
01		a, b, c	
.			
.			
.			
.			

Régions existants

Les identificateurs des images qu'elles sont pour la région 01 une valeur b

Exemple 3.4 :

Pour les images de la Figure 3.9 avec des paramètres de découpage on aura l'arbre quaternaire représenté dans la Figure 3.10 :

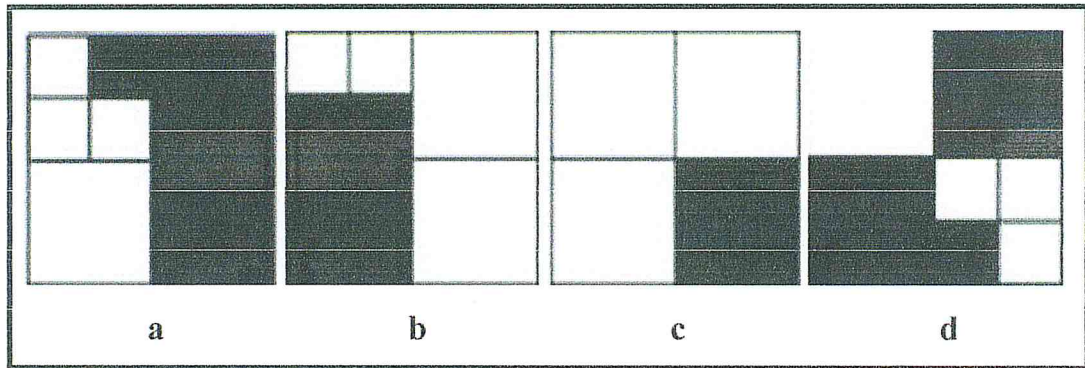


Figure 3.9 – Ensemble des images.

Regions	Noir	Blanc	Int
00		c,d	a,b
01	a,d	b,c	
02	b,d	a,c	
03	a,c	b	d
000		a,b	
001	a	b	
002	b	a	
003	b	a	
030		d	
031		d	
...			

Figure 3.10 –présentation de l'arbre quaternaire générique SPI

Graphe de partage

Dans notre travail on calcule pour toutes couples d'images dans notre base d'images la distance. de cette manière on obtient un graphe qui a pour sommets les images reliées par des arcs correspondant à la distance de chaque deux image.

Le graphe correspondant aux images de figure 3.9 est représenté dans la Figure 3.11

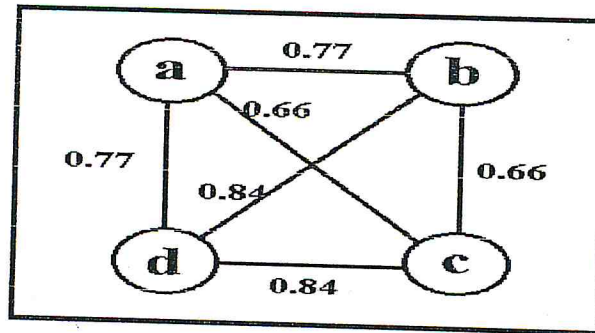


Figure 3.11 –graphe de partage

Déduire l'arbre couvrant minimum

Dans le chapitre précédant on a dit qu'il faut déduire l'arbre quaternaire maximum alors que dans notre travail on doit calculer l'arbre couvrant minimum parce que l'arc de graphe contient une distance et on s'intéresse au cas où les distances sont petites.

Pour déduire l'arbre couvrant on doit pour chaque image choisir l'arc de poids minimum.

Déduire l'arbre couvrant minimum de la Figure 3.11

Les distances existant :

$$D(a,b)=0,77 \quad D(a,c)=0,66 \quad D(a,d)=0,77 \quad D(b,c)=0,66 \quad D(b,d)=0,84 \quad D(c,d)=0,84$$

On commence par l'image "a":

$$D(a,c) < D(a,b) \leq D(a,d) \rightarrow \text{on prend l'arc (a, c)}$$

En suit l'image "c" avec les autre images:

$$D(c,b) < d(c, d) \rightarrow \text{on prend l'arc (c, b)}$$

En suit l'image "b" avec image d :

$$D(b, d) \rightarrow \text{on prend l'arc (b, d)}$$

A la fin on a l'arbre couvrant minimum :

$$a \rightarrow c \rightarrow b \rightarrow d$$

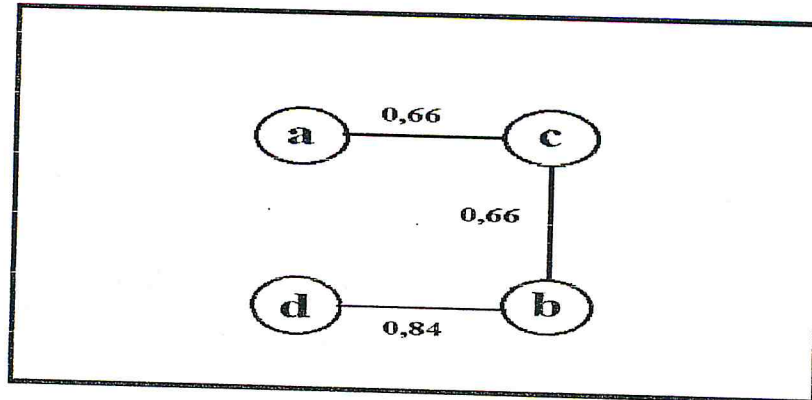


Figure 3.12 -L'arbre couvrant minimum de la figure 3.10

Racine de l'arbre couvrant minimum

Pour trouver la racine de l'arbre couvrant minimum on calcule le profond pour chaque arbre possible et on choisit l'arbre qui a la profondeur pondérée la plus petite.

La profond pondérée* de l'arbre :c'est le maximum des sommes des arcs de chaque chemin de puis la racine.

Pour la Figure 3.12 les arbres possibles sont :

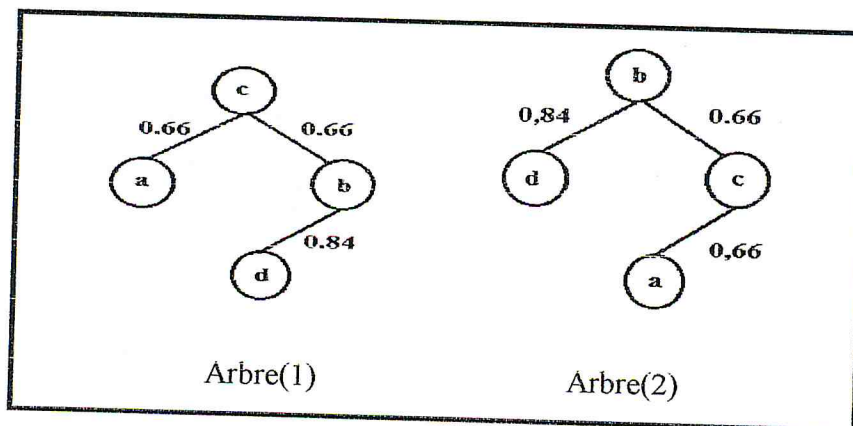


Figure 3.13 – L'arbre possible pour la Figure 3.12

Profond de l'arbre (1) égale 1,5.

Profond de l'arbre (2) égale 1,32.

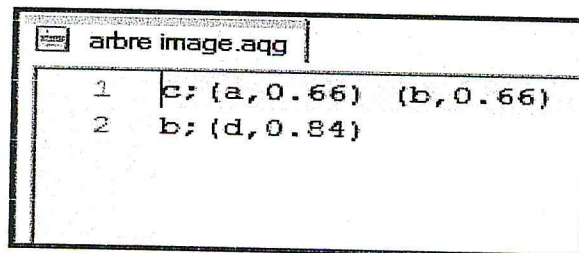
On prend l'arbre de petit profond (l'arbre (2)).

Stockage de l'arbre d'images

On a proposé une manière de stocker l'arbre d'images dans un fichier d'extension (.aqg) de manière qu'on peut récupérer l'arbre d'image.

Notre fichier contient des lignes commencent par des racines suivies de files, avec les distances correspondantes et les autres lignes contiennent des nœuds père avec leurs fils avec les distance correspondant.

Fichier de stockage pour l'arbre (1) de Figure 3.13 :



arbre image.aqg	
1	c; (a, 0.66) (b, 0.66)
2	b; (d, 0.84)

Figure 3.14 – fichier de stockage de l'arbre d'images.

Construire l'arbre quaternaire générique final

Pour construire l'arbre quaternaire générique on utilise la méthode qu'on a expliqué dans le chapitre deux.

Pour cela on utilise l'arbre d'images et l'arbre quaternaire générique SPI pour construire l'arbre quaternaire générique.

Affichage de l'arbre quaternaire générique des images de la figure 3.9 déduire l'arbre à partir de arbre quaternaire générique spi présenté dans la figure 3.10 et l'arbre d'images (1) de la Figure 3.13 :

Regions	Noir	Blanc	Int
00		a,d	c
01	c	a,b	
02	b,d	c	
03	c	b	d
000		c	
001	c	b	
002	b	c	
003	b	c	
030		d	
031		d	

Figure 3.15 _ l'arbre quaternaire générique final.

Les images en couleur

Dans les images en couleur il y a des difficultés et des problèmes difficiles à résoudre, parmi ces problèmes la valeur associée à une région. Dans les images noir et blanc c'est facile soit noir soit blanc mais dans les images couleur c'est différent, puisque une image couleur est composée de trois couches de couleur on parle d'espace couleur RGB ou RVB (rouge, vert, bleu) (Figure 3.16).

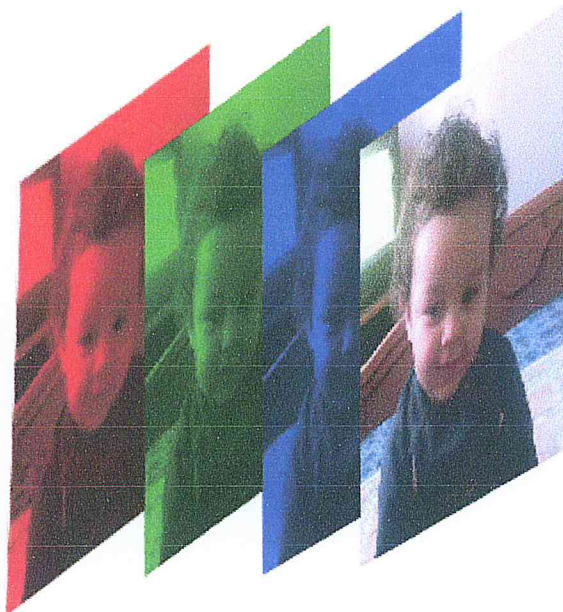


Figure 3.16 – Image en couleur composée de trois couches

Les pixels de chaque couche ont une valeur entre 0 et 255.

Les pixels de l'image couleur sont représentés par 3 valeurs des couches par exemple

Pour un pixel de couleur rouge est (255, 0,0).

Donc notre espace couleur a $255*255*255=16\ 581\ 375$ couleurs.



La difficulté qui se pose c'est que dans notre nœud d'arbre d'image on peut avoir 16 581 375 valeurs différentes, ce nombre est vraiment grand pour qu'on peut le manipuler, il y aura des problèmes dans le stockage et le calcul va prendre beaucoup de temps.

Ce problème il reste le plus difficile à résoudre dans l'arbre quaternaire générique pour les images en couleurs.

Solution proposée :

Pour diminuer l'espace couleur on a choisi de calculer le pourcentage de chaque couche de couleur, de ça la région aura une valeur qui se compose de trois pourcentages.

Comment calculer les pourcentages ?

Pour la couleur rouge :

$$prR = \frac{\sum \text{valeurs des pixels de couche rouge}}{\sum \text{valeurs des pixels des trois couches}}$$

Pour la couleur vert :

$$prV = \frac{\sum \text{valeurs des pixels de couche vert}}{\sum \text{valeurs des pixels des trois couches}}$$

Pour la couleur bleu :

$$prB = \frac{\sum \text{valeurs des pixels de couche bleu}}{\sum \text{valeurs des pixels des trois couches}}$$

Maintenant notre région est représentée par trois pourcentages (prR , prV , prB) c'est à dire on a $100*100*100=1000000$ valeurs possibles, ce nombre est plus petit que 16 581 375 mais il reste un grand nombre, donc on a découpé l'intervalle

de pourcentage $[0,100]$ en quatre intervalles et les valeurs de chaque intervalle prend la valeur de son intermédiaire. (Figure 3.17).

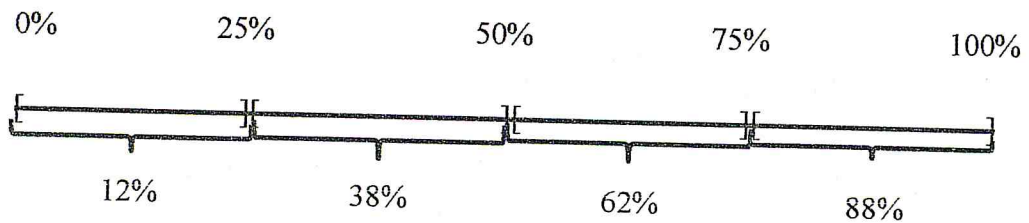


Figure 3.17 – découpage de l'intervalle

L'Intermédiaire d'un intervalle est la somme de deux bornes de l'intervalle sur deux.

De cette manière les pourcentages de l'intervalle $[0,25]$ prennent la valeur 12, pour les pourcentages de l'intervalle $[25,50]$ prennent la valeur 38, les pourcentages de l'intervalle $[50,75]$ prennent la valeur 62 et les pourcentages de l'intervalle $[75,100]$ prennent la valeur 88.

Maintenant la région est représenté par trois valeurs dont chaque valeur prend quatre valeurs possible comme par exemple $(12, 38, 38)$, cela donne $4 \times 4 \times 4 = 64$ valeurs possible, mais il y aura des valeurs interdit ou faux comme $(88, 88, 88)$ cela signifie que le rouge doit être supérieur de 75% et vert aussi et bleu aussi et cela ce n'est pas vrais pacque la somme de pourcentage est supérieur à 100% alors on élimine cette combinaison.

Après élimination de combinaisons fausse on aura 13 combinaisons valides :

$(12, 12, 38)$, $(12, 12, 62)$, $(12, 12, 88)$, $(12, 38, 12)$, $(12, 38, 38)$, $(12, 62, 12)$, $(12, 88, 12)$,

$(38, 12, 12)$, $(38, 12, 38)$, $(38, 38, 12)$, $(38, 38, 38)$, $(62, 12, 12)$, $(88, 12, 12)$.

Les régions prennent une de ces valeurs, ce que donne notre arbre quaternaire générique ; le nœud aura 13 valeurs au maximum et si on Projette ça sur la

probabilité ça donne que dans 14 régions différentes on a forcément deux régions de même valeur.

Le processus de construire de l'arbre quaternaire générique est représenté dans la Figure 3.18.

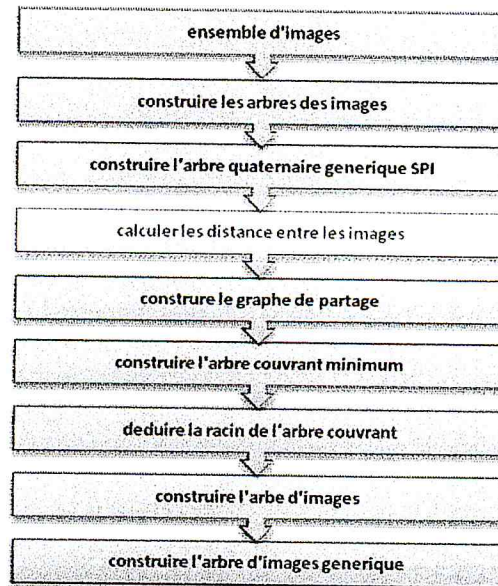


Figure 3.18 –processus de construction de l'arbre générique pour les images couleur

Construire les arbres des images :

Critères de découpage

Niveau de découpage

Découper l'image jusqu'à on atteindre le niveau définie.

Domination de couleur

Seuil de découpage

Dans notre travail les seuils prennent un de ces quatre valeurs (12, 38, 62,88), si on dit qu'une valeur égale au seuil 12 c'est-à-dire elle est dans l'intervalle [0,25]

et si on dit qu'elle est supérieure à 12 ça signifie qu'elle est dans l'intervalle [25,100].

Critère égale au pourcentage

On calcule les pourcentages prR , prV , prB puis on définit un seuil, si l'un des trois égales au seuil on s'arrête Le découpage et on associe la combinaison (prR , prV , prB) à la région.

Critère égale au pourcentage avec un niveau

On combine les deux critères niveau de découpage et égale de pourcentage.

Critère supérieur ou égal pourcentage

On calcule les pourcentages prR , prV , prB puis on définit un seuil, si l'un des trois est égal ou supérieur du seuil on arrête le découpage et on associe la combinaison (prR , prV , prB) à la région.

Critère supérieur ou égale au pourcentage avec un niveau

On combine les deux critères « niveau de découpage » et le critère « supérieur ou égale pourcentage ».

Critère égale à la combinaison

On calcule les pourcentages prR , prV , prB puis on définit une combinaison R , V , B , si $prR=R$, $prV=V$, $prB=B$ on arrête le découpage et on associe la combinaison (prR , prV , prB) à la région.

Critère égale à la combinaison avec un niveau

On combine les deux critères niveau de découpage et égale à la combinaison pourcentage.

Critère Homogénéité

Dans notre travail on a utilisé une méthode pour savoir si la région est homogène ou pas.

Explication de la méthode :

On définit trois valeurs d'homogénéité HR, HV, HB et un pourcentage d'homogénéité PH.

PR=Valeur de pixel dans la couche rouge.

PV=Valeur de pixel dans la couche vert.

PB=Valeur de pixel dans la couche bleu.

PR(p)=Valeur de pixel voisin de pixel p dans la couche rouge.

PV(p)=Valeur de pixel voisin de pixel p dans la couche vert.

PB(p)=Valeur de pixel voisin de pixel p dans la couche bleu.

Pour chaque voisin de pixel p de l'image :

Si

$$PR-HR \leq PR(p) \leq PR+HR \text{ ou}$$

$$PV-HV \leq PV(p) \leq PV+HV \text{ ou}$$

$$PB-HB \leq PB(p) \leq PB+HB$$

On augmente le variable ho si non on augmente le variable nho, à la fin on calcule pourcentage de homogénéité qui égale :

$Pr = ho / (ho + nho)$;

Ensuite si Pr est supérieur ou égal à PH on arrête le découpage sinon on découpe la région.

Critère homogénéité avec niveau

On combine les deux critères niveau de découpage et le critère homogénéité.

Distance utilisé

On a utilisé la distance de Q-similarité qu'on a expliqué dans le chapitre 2

La distance de Q-similarité des images i et i' notée $d(i, i')$, est calculée par l'équation suivante :

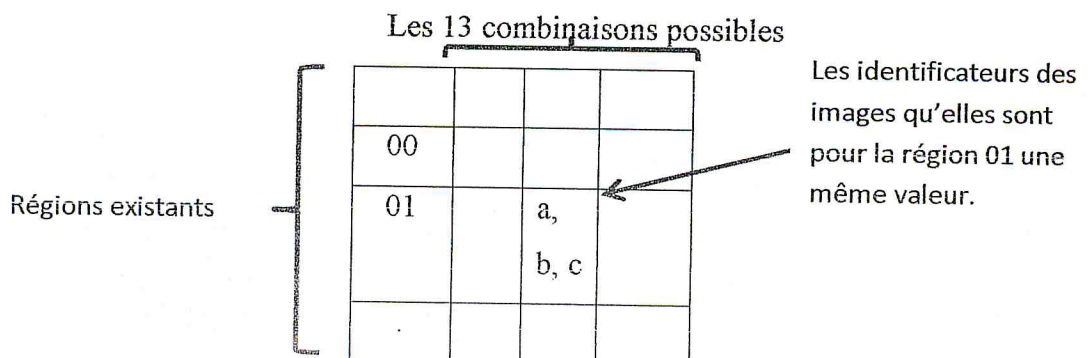
$$D(i, i') = \frac{|S(i, i')|}{|U(i, i')|} = \frac{\text{nombre de nœuds différents}}{\text{Total des identificateurs de nœud (sans doublon)}}$$

Construire l'arbre quaternaire générique spi

On utilise les arbres d'images pour construire l'arbre spi de la même manière qu'on a expliqué dans le chapitre 2.

Chaque nœud de l'arbre contient le nom de région et une valeur de 13 combinaisons possible associées avec des identificateurs des images et une valeur Int associé avec des identificateurs des images.

On a proposé une représentation de l'arbre par une matrice comme ce si :



Exemple 3.6 :

Pour les images de la Figure 3.19 avec des paramètres de découpage on aura l'arbre quaternaire représenté dans la Figure 3.20 :

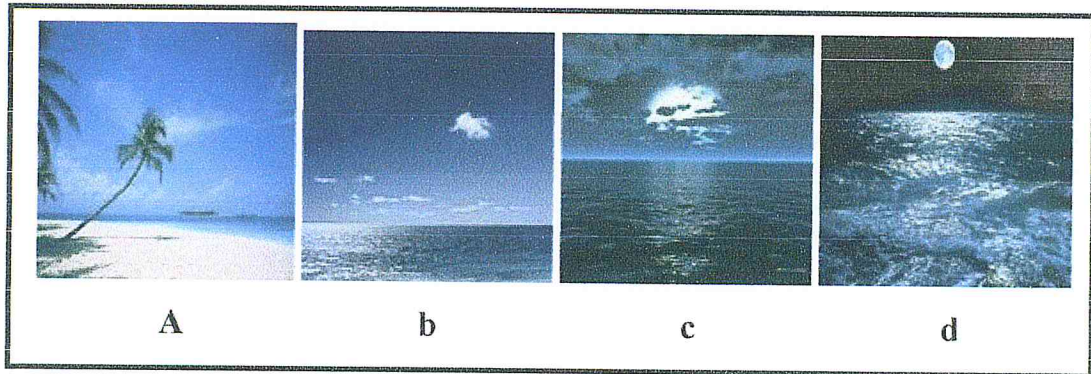


Figure 3.19 – Ensemble des images.

Regions	R=[0,25],V=[0,25] B=[25,50]	R=[0,25],V=[0,25] B=[50,75]	R=[0,25],V=[0,25] B=[75,100]	R=[0,25] B=[0,25]
00				
01				
02				
03				
000		a		
001		a		
002		a		
003		a		
010		a		
011		a		

Figure 3.20 –présentation de l'arbre quaternaire générique SPI

Graphe de partage

Dans notre travail on calcule pour toutes couples d'images dans notre base d'images la distance. de cette manière on obtient un graphe qui a pour sommets les images reliées par des arcs correspondant à la distance de chaque deux image.

Le graphe correspondant a les images de figure 3.19 est représenté dans la Figure 3.21

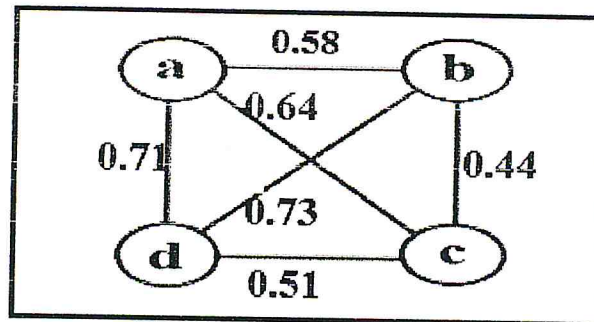


Figure 3.21 – graphe de partage

Déduire l'arbre couvrant minimum

Dans le chapitre précédant on a dit qu'il faut déduire l'arbre quaternaire maximal alors que dans notre travail on doit calculer l'arbre couvrant minimum qui sert à calculer les petites distances.

Pour déduire l'arbre couvrant on doit pour chaque image on choisit l'arc de poids minimum.

Déduire l'arbre couvrant minimum de la Figure 3.21

Les distances existantes :

$$D(a,b)=0,58 \quad D(a,c)=0,64 \quad D(a,d)=0,71 \quad D(b,c)=0,44 \quad D(b,d)=0,73 \quad D(c,d)=0,51$$

On commence par l'image "a":

$$D(a,b) < D(a,c) \leq D(a,d) \rightarrow \text{on prend l'arc (a, b)}$$

Profond de l'arbre (1) égale 1,02.

Profond de l'arbre (2) égale 0,95.

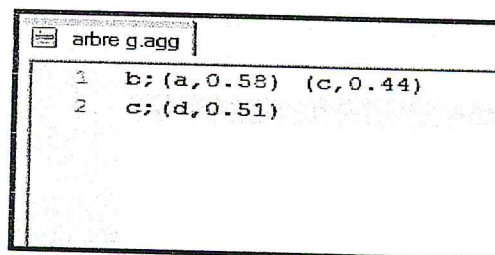
On prend l'arbre de petit profond (l'arbre (2)).

Le Stockage de l'arbre d'images

On a proposé une manière de stocker l'arbre d'images dans un fichier de extension (.agg) de manier que on peut récupérer l'arbre image.

Notre fichier est composé des lignes, la premier ligne contient au début la racine en suit ces fils avec les distance correspondant et les autres lignes contient les nœuds père avec leurs fils avec les distance correspondant.

Fichier de stockage pour l'arbre (2) de Figure 3.13 :



```
arbre g.agg
1 b; (a, 0.58) (c, 0.44)
2 c; (d, 0.51)
```

Figure 3.24 – fichier de stockage de l'arbre d'images.

Construire l'arbre quaternaire générique final

Pour construire l'arbre quaternaire générique on utilise la méthode qu'on a expliqué dans chapitre deux.

Pour cela on utilise l'arbre d'images et l'arbre quaternaire générique SPI pour construire l'arbre quaternaire générique.

Affichage de l'arbre quaternaire générique d'un ensemble d'images Figure 3.25 :

$R=[25,50]$	$R=[25,50], V=[25,50]$ $B=[25,50]$	$R=[50,75], V=[0,25]$ $B=[0,25]$	$R=[75,100], V=[0,25]$ $B=[0,25]$	Int
				d
				d
				d
				d
				d
				d
				d
				d
				d
				d
				d
				d

Figure 3.25 - l'arbre quaternaire générique final.

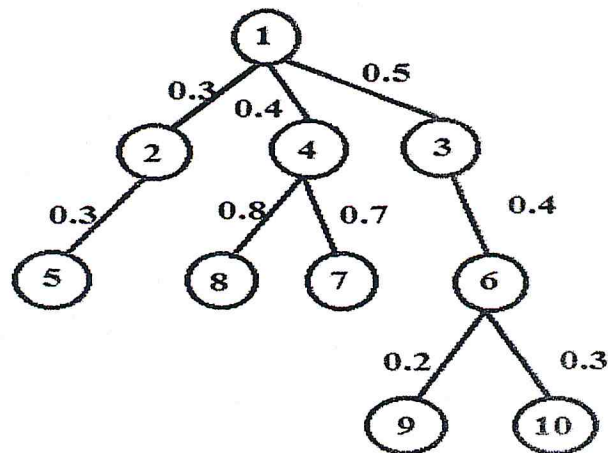
IV. Recherche des images similaires dans l'arbre d'images

Après construire l'arbre d'images on l'utilise pour faire des requête image sur l'arbre pour choisit les images similaires.

Déroulement de recherche :

On calcule la distance entre l'image requête et la racine en suit on le compare avec les distance des fils de racine le fils le plus proche à notre distance on prend et on calcule la distance entre l'image requête et ce fils en suit on compare avec ses fils jusqu'à la fin de notre arbre.

Exemple 3.7:



On calcule la distance entre l'image requête et l'image 1 par exemple on a la distance égale 0.6 ,la distance le plus poche à cette distance c'est la distance entre l'image 1 et 3 alors on prend l'image 3 en suit on prend l'image 6 en suit on calcule la distance entre l'image requête et l'image 6 par exemple on a 0.4 la distance le plus proche à cette distance c'est la distance entre l'image 6 et 10 alors on prend l'image 10 .

Les images les plus similaires à notre requête sont les images 3, 6,10.

Conclusion

Dans ce chapitre, nous avons présenté la conception et la mise en œuvre de notre système d'indexation et de la recherche d'images en utilisant les arbres quaternaires. Nous avons pu avoir une idée sur l'impact du choix du critère sur la nature du résultat. Nous avons laissé à l'utilisateurs la possibilité d'afficher sa recherche eu fonction de ses besoin.

Implémentation et résultats expérimentaux

I. Introduction

Dans ce chapitre, nous allons décrire l'implémentation de notre logiciel. On propose par la suite une étude des divers tests, ainsi que les résultats obtenus.

Langage utilisé : on a travaillé avec java sous plateforme netbeans 6.9.

II. Présentation du logiciel

Nous allons présenter la fenêtre principale de l'application et les différents outils permettant la communication avec le logiciel. La Figure 4.1 montre la structure de la fenêtre principale.

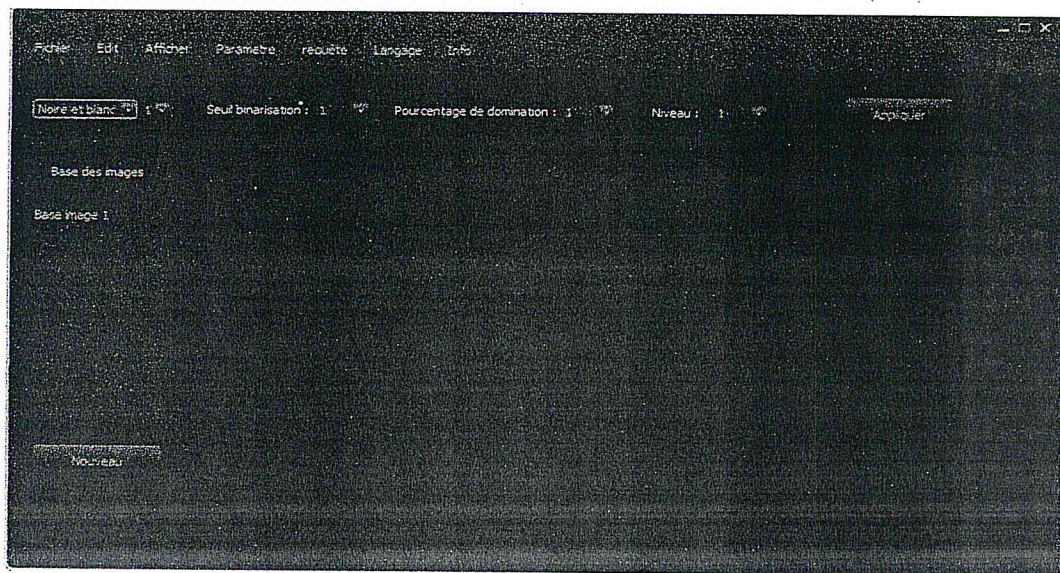


Figure 4.1-fenêtre principale.

Menu/Fichier :

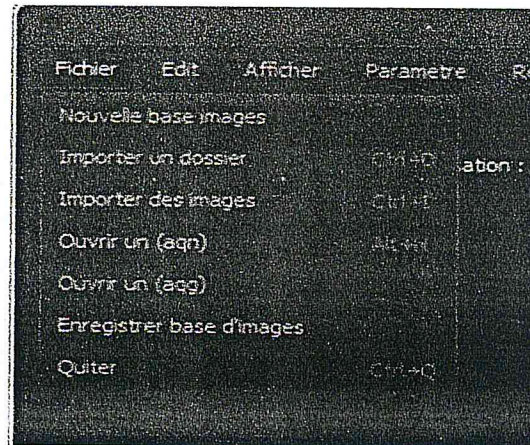


Figure 4.2 Menu/Fichier

Nouvelle base image: permettre de crée une nouvelle base d'images.

Importer dossier : permettre d'importer un dossier d'images.

Importer images : permettre d'importer des images.

Ouvrir un (aqn) : permettre d'afficher l'arbre d'une image stocker dans un fichier (aqn).

Ouvrir un (aqg) : permettre d'afficher l'arbre d'images stocker dans un fichier (aqg).

Enregistrer base images : permettre de stocker l'arbre d'images.

Quitter : permettre de quitter logiciel.

Menu/Edit :

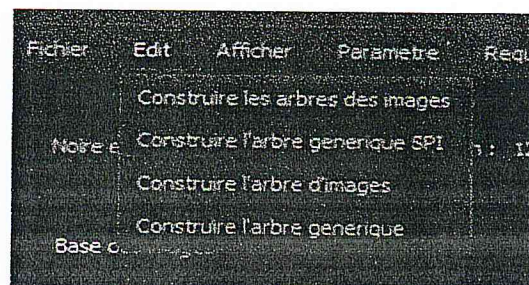


Figure 4.3 Menu/Edit.

Construire les arbres des images : permettre de créer les arbres des images de la base d'images.

Construire l'arbre générique spi : permettre de créer l'arbre générique spi.

Construire l'arbre d'images : permettre de créer l'arbre d'images.

Construire l'arbre générique : permettre de créer l'arbre générique.

Menu/Afficher :

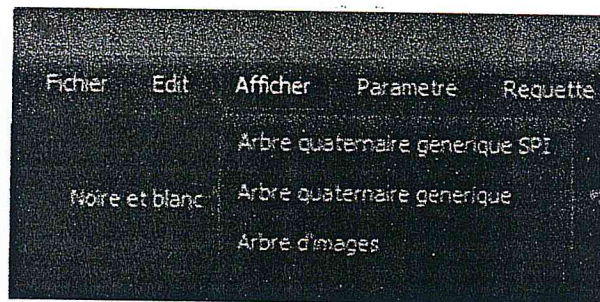


Figure 4.4 Menu/Afficher

Arbre quaternaire générique spi : permettre d'afficher l'arbre spi.

Arbre quaternaire générique : permettre d'afficher l'arbre générique.

Arbre d'images : permettre d'afficher l'arbre d'images.

Menu/Paramètre :



Figure 4.5 Menu/Paramètre.

Mode : permettre de changer entre mode noir et blanc et couleur.

Paramètres globale : permettre de paramétrer toutes les paramètres d'une seule fenêtre.

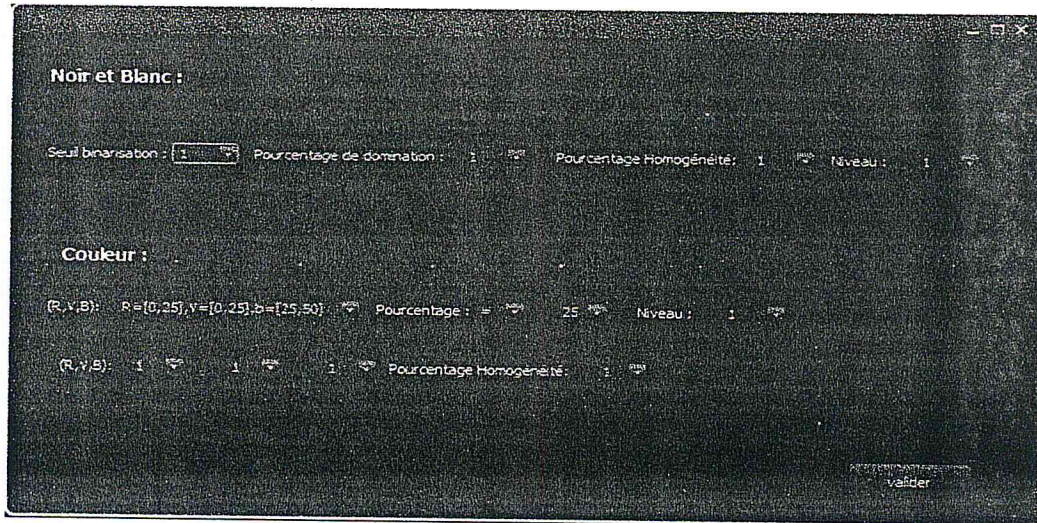


Figure 4.6 - Menu/Parametre/parametres globale.

Menu/Requête :



Figure 4.7 Menu/requête.

Requête : permettre de lancer une recherche par requête image.



Figure 4.8 fenêtre de requête.

Menu/langage : permettre de choisi langage de fenetre.



Figure 4.9 menu/langage.

Menu/Info:

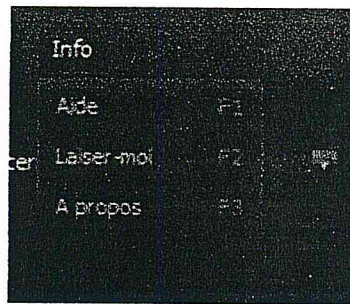


Figure 4.10 menu/Info.

Jcombox dans la figure 4.11 permettre de choisi le mode d'image noir et blanc ou couleur.

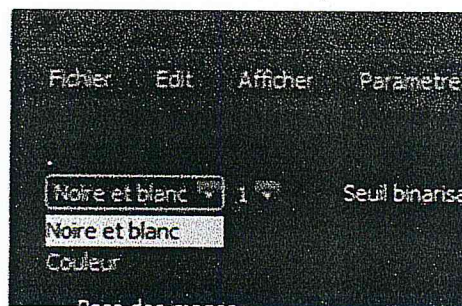


Figure 4.11 jcombox de mode.

Paramètres de mode noir et blanc (couleur dominante) :



Figure 4.12 les défèrent paramètres de couleur dominante de mode noir et blanc.

Bouton appliquer : permettre d'appliquer les paramètres définis a toutes les images dans la base.

Paramètres de mode noir et blanc (homogénéité) :

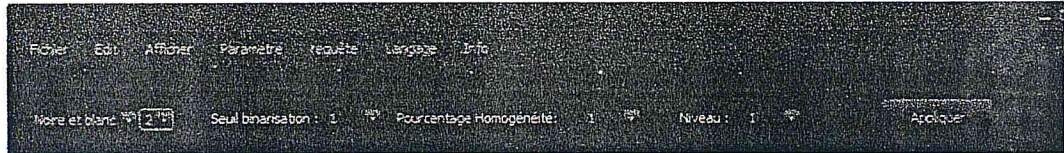


Figure 4.13 les défèrent paramètres de homogénéité de mode noir et blanc.

Paramètres de mode couleur (couleur dominante) :



Figure 4.14 les défèrent paramètres couleur dominante de mode noir et blanc.

Paramètres de mode couleur (homogénéité) :

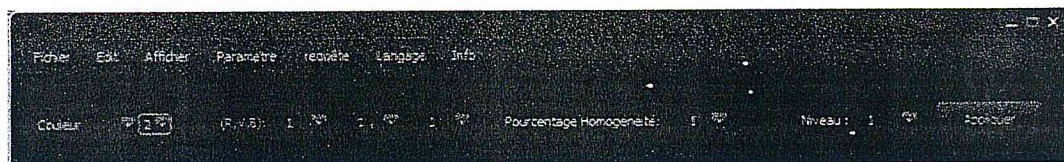


Figure 4.15 les défèrent paramètres homogénéité de mode couleur.

III. Les Teste :

III.1. Les images noir et blanc

a) Stockage d'image

On a appliqué le critère de domination de couleur sur l'image de Figure 4.16 avec les paramètres suivant :

Seuil de binarisation= 128.

Pourcentage de domination = 70%.

Niveau de découpage= 4.

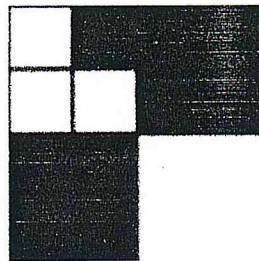


Figure 4.16 –Image noir et blanc.

L'affichage de fichier de stockage donne :

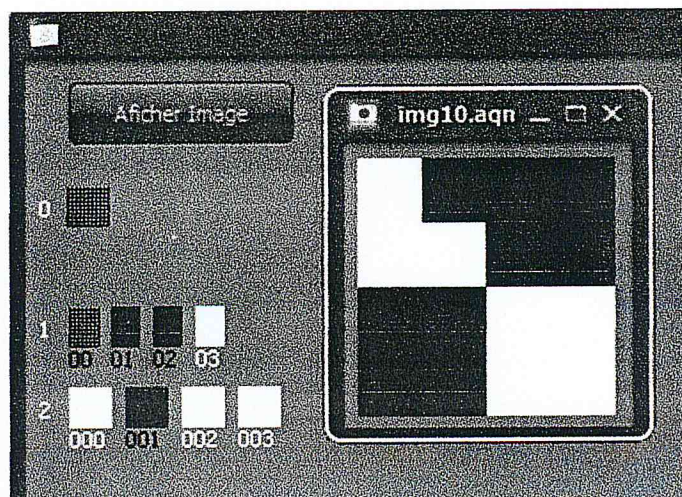


Figure 4.17 –L'affichage de fichier (.aqn)

Maintenant on applique les meme parametre sur l'images de Figure 4.18 :



Figure 4.18 –Image noir et blanc

L'affichage de fichier donne :

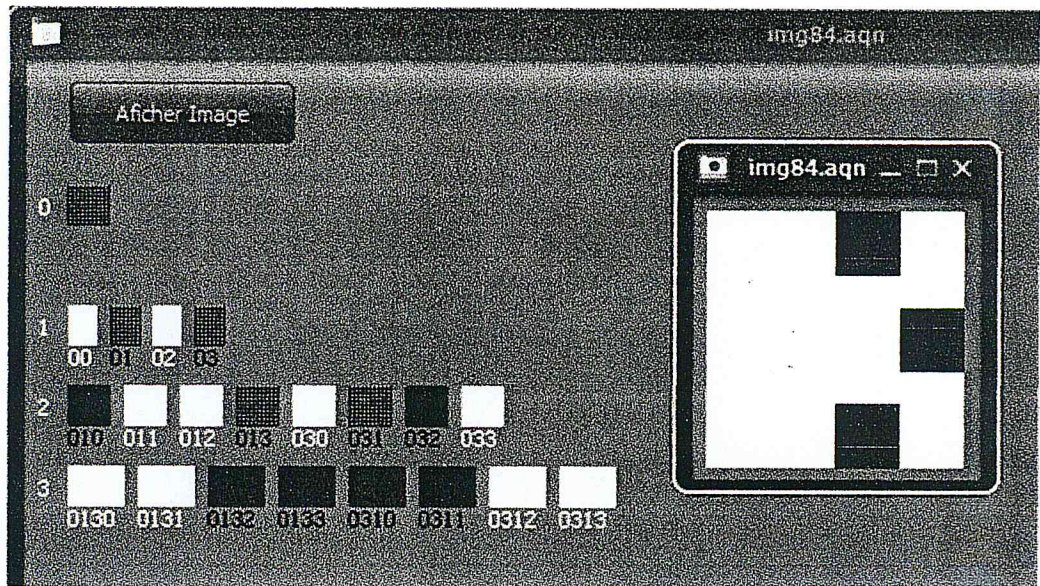


Figure 4.19- L'affichage de fichier (.aqn)

On remarque que l'image perdre beaucoup details car les parametres definie ils sont pas bien precisi pour donner des bon resultat,maintenant on va l'appliquer d'autres parametre sur l'image :

Seuil de binarisation= 128.

Pourcentage de domination = 90%.

Niveau de decoupage= 10.

Le résultat d'affichage donne :



Figure 4.20 L'affichage de fichier (.aqn)

La taille de notre image est 24ko et la taille de notre fichier (.aqn) est 19ko on peut dire qu'on a gagné 5ko d'espace.

b) Recherche d'une image dans l'arbre d'images

On applique les paramètres suivants sur les images dans la Figure 4.21 pour construire l'arbre d'images :

Seuil Binarisation= 128.

Domination= 60%.

Niveau= 4.

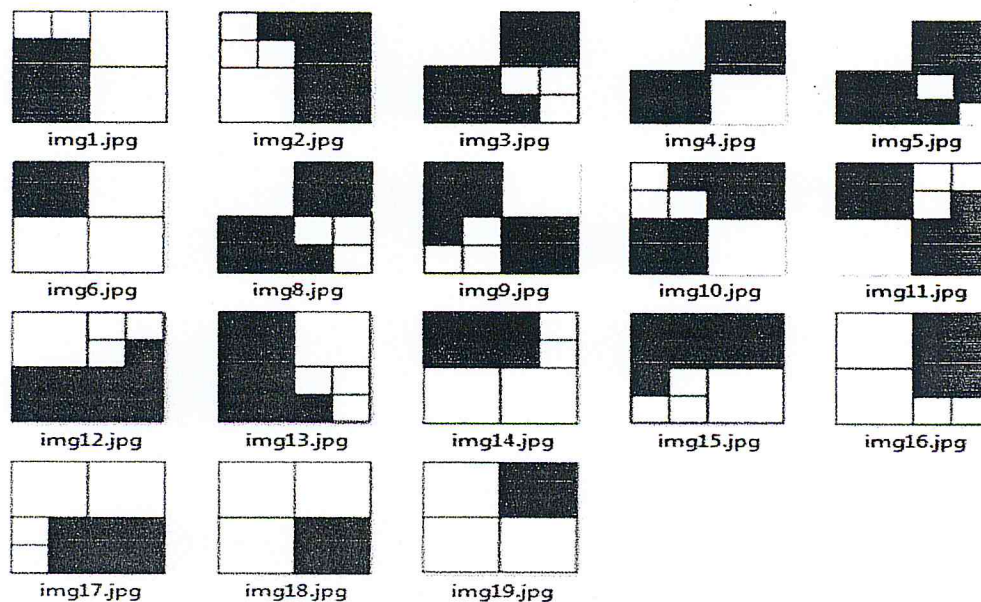


Figure 4.21-base d'images

En suit on fait un recherche avec la requete d'image de Figure 4.22 :

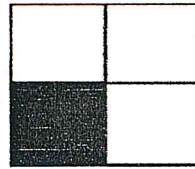


Figure 4.22- Image requete.

Resultat de recherche est presenté dans la Figure 4.23 :

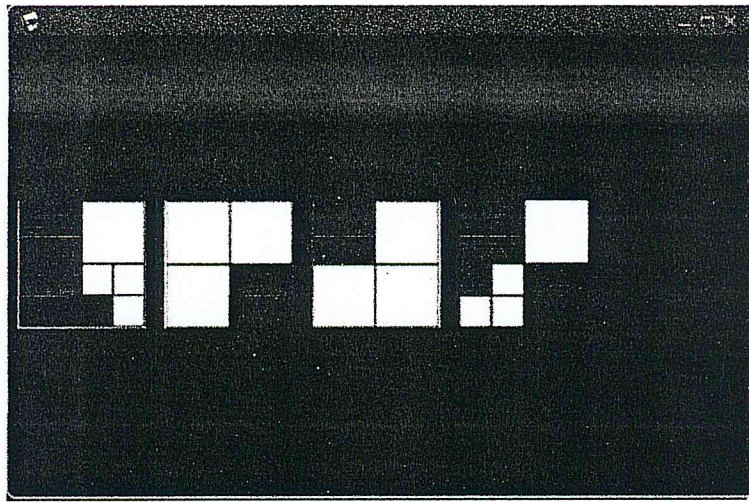


Figure 4.23-resultat de recherche.

Discussion sur resultat :

Avec les parametres definie on aura pour notre image reque les region suivant :

00=blanc, 01=blanc, 02=noir, 03=blanc.

Pour la premier image de resultat on a les region suivant :

00=noir, 01=blanc, 02=noir, 03=blanc.

On remarque que l'image requete et la premier image de resultat ont des regions commun (01,02,03) et une region defirente (00) ce que donne que les deux image sont similaire.

Meme chose pour la deuxieme image on a les regions :

00=blanc, 01=blanc, 02=blanc, 03=noir.

Elle a deux regions comun avec l'image requete (00,01).

III.2 Les images couleur

On a appliquer les parametre suivant sur les images de Figure4.24 :

Critère domination couleur de pourcentage supérieur ou égale =50.

Niveau=4.

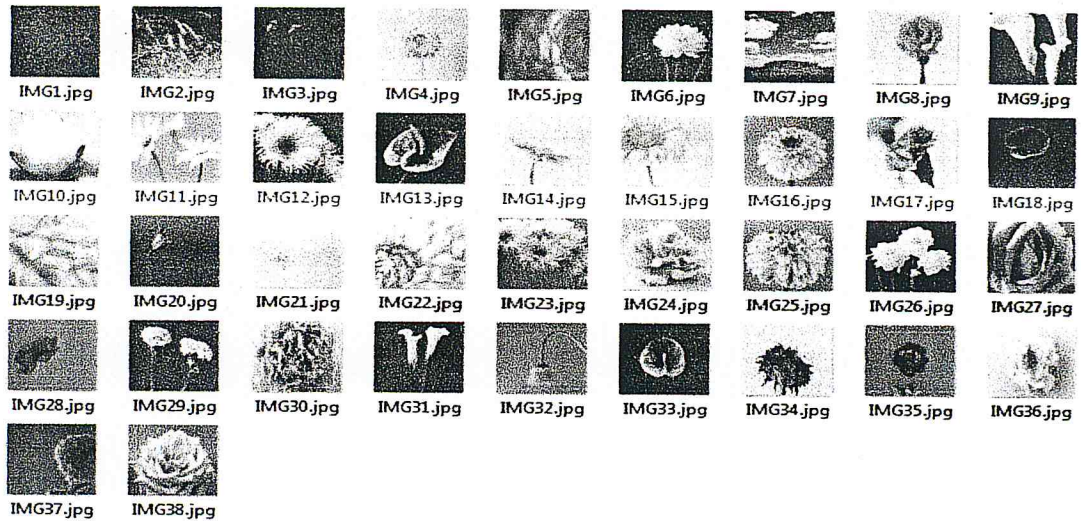


Figure 4.24-base d'images couleur.

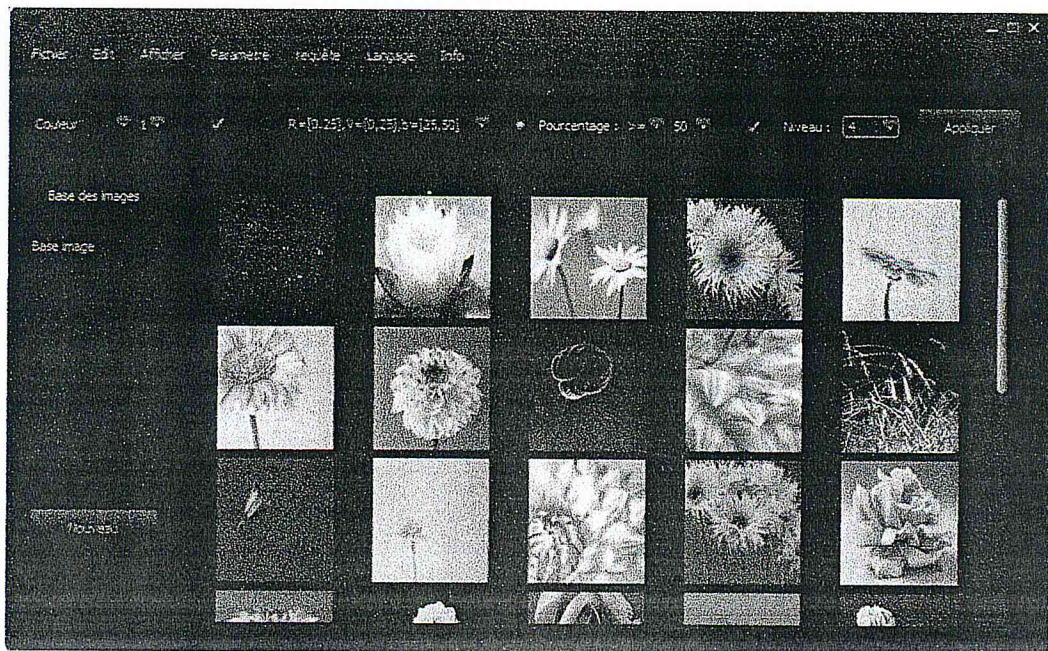


Figure 4.25-parametres utilisé.

En suit on fait un recherche avec la requete d'image de Figure 4.26 :

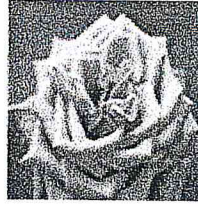


Figure 4.26- Image requête.

Resultat de recherche est presenté dans la Figure 4.27 :

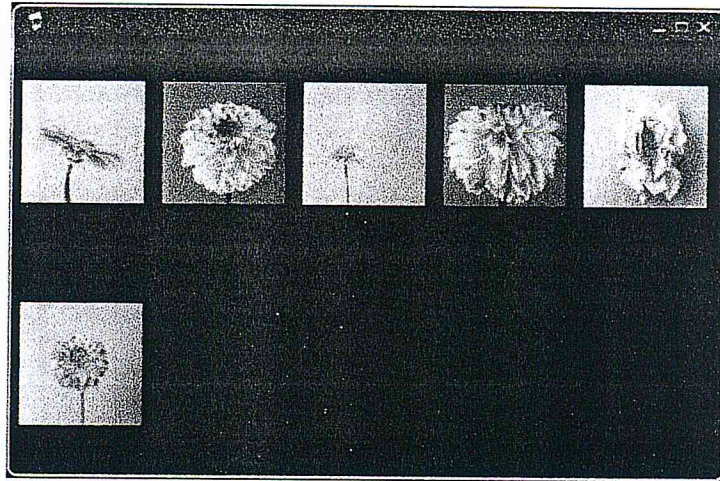


Figure 4.27- Résultat de recherche

On applique d'autres paramètres sur les images avec la même image requête:

Critère égale à la combinaison = (12, 12,38) ou bien

Rouge $\in [0,25]$, Vert $\in [0,25]$, Bleu $\in [25,50]$,

Niveau=4.

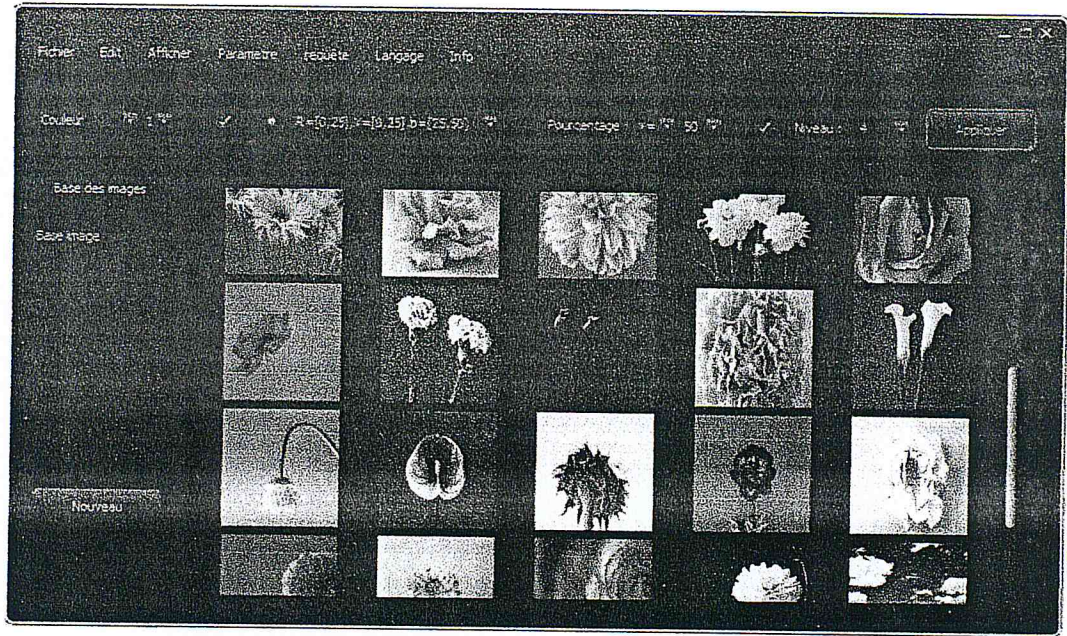


Figure 4.28- paramètres de recherche.

Resultat de recherche est présenté dans la Figure 4.29 :

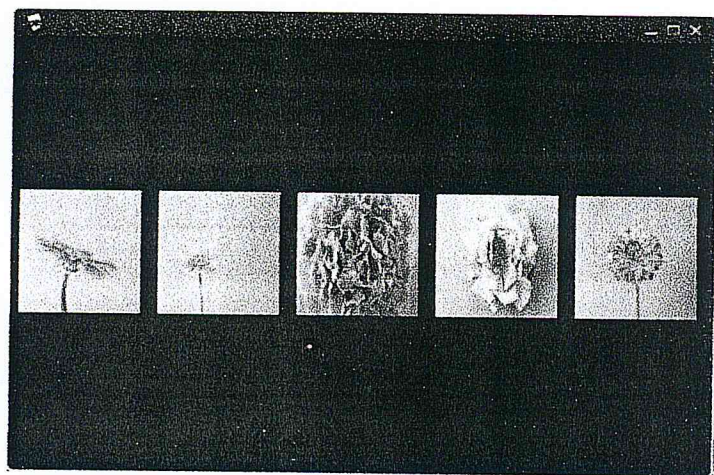


Figure 4.29 - Résultat de recherche.

On applique d'autre paramètres sur les images avec la même image requête:

Critère homogénéité : $R=8$, $V=10$, $B=6$, pourcentage de homogénéité=90%.

Niveau=4.

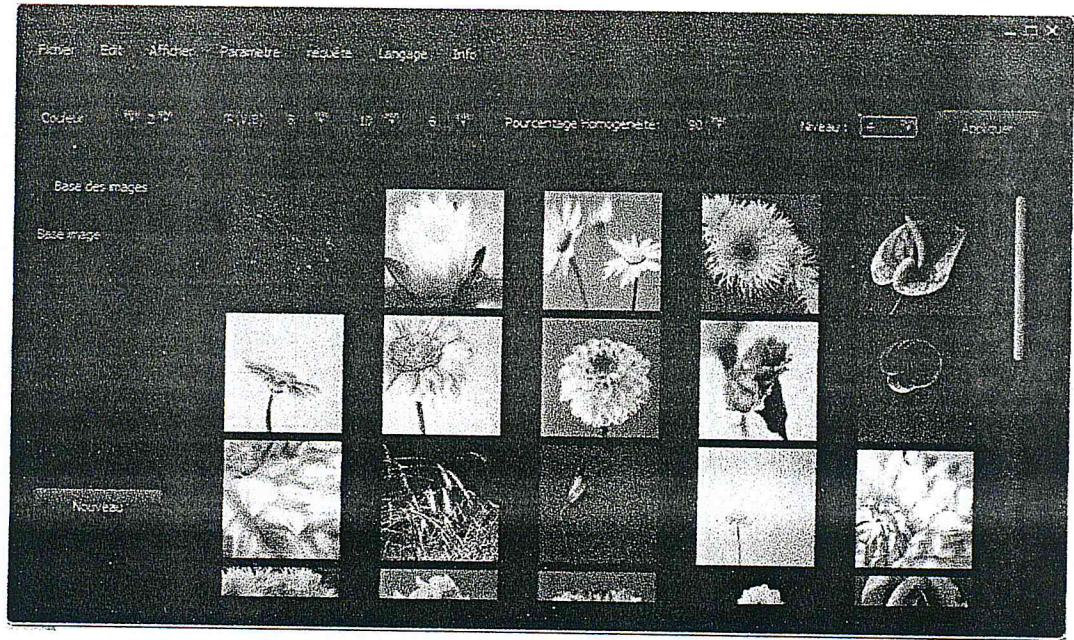


Figure 4.30- paramètres de recherche.

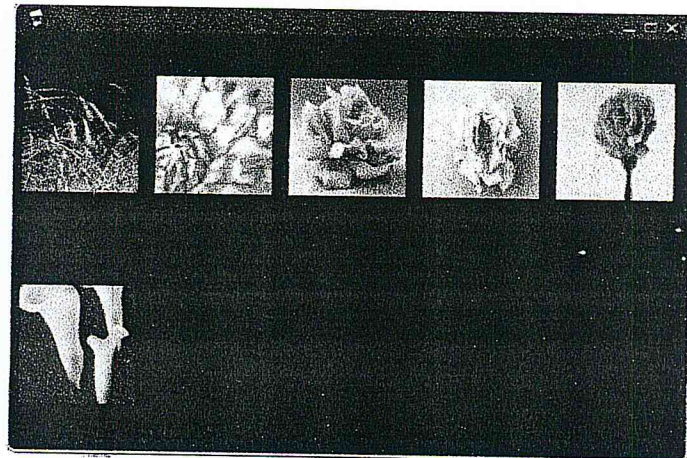


Figure 4.31- Résultat de recherche.

CONCLUSION

Tout au long de ce chapitre, on a essayé de voir tous les résultats qui peuvent être donnée en fonction des critères que ça soit entre les images noir et blanc ou bien les images couleur.

Conclusion générale

Plusieurs études traitent de la recherche d'images par leur contenu. En effet, l'image comprend une quantité importante d'information que l'homme utilise pour l'identifier et la décrire. Une description textuelle n'est donc pas toujours utile ni facile à déployer. Dans ce cadre, au lieu d'indexer les images par une description textuelle (mots), on se dirige vers sa description par son contenu.

Pour ce faire, on doit au préalable extraire des indices visuels de l'image. Qu'on a représenté par les arbres des images qui permettront d'indexer l'image par son contenu.

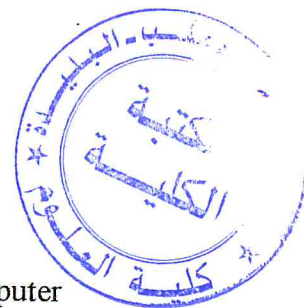
Notre travail et d'organiser les images sous forme d'un arbre quaternaire générique et de l'utiliser pour la recherche d'image, notre travail s'inscrit dans ces méthodes de recherche d'image par le contenu (dénommées content-based image retrieval).

Le processus de recherche s'effectue alors, non pas à partir d'une requête textuelle, mais à partir d'une image requête pour laquelle les paramètres de couleur, sont calculés. Les images les plus similaires à l'image requête sont présentées en résultat à l'utilisateur comme étant les images les plus proche visuellement de l'image requête.

Les tests réalisés donne une idée sur la qualité de ce travail. Tout fois beaucoup plus de tests et des basse d'images plus étoffée permettront de mieux apprécier la qualité du travail réalisé.

Ce travail pu d'être enrichi par les critères forme, texture. Il mérite d'être mis à l'épreuve sur des bases servant à des produits comme QBIC, pour estimer comme de ce modeste travail.

Webographies



- [1] Swain M., Ballard D., "Color Indexing", International Journal of computer Vision, Vol. 7, n°1, p. 1-13, 1991.
- [2] M. Stricker, A Orengo, "Similarity of color images", In Proc. SPIE Storage and Retrieval for Image and Video Databases III, Vol 2420, pp. 381-392, February 1995.
- [3] Haralik, R. M., Shanmugam, K., et Dinstein, I. "Textural features for Images Classification". IEEE Transaction on System, Man, Cybernetics, 3, 610-621, 1973.
- [4] Daoudi Imane these de doctorat . "Recherche par similarité dans les bases de données multimédia : application à la recherche par le contenu d'images", NIVERSITÉ MOHAMMED V – AGDAL, FACULTÉ DES SCIENCES Rabat, 17 Juillet 2008.
- [5] Houaria ABED, Lynda ZAOUI , Université des Sciences et de la Technologie d'Oran - Mohamed Boudiaf , " Système D'Indexation et de Recherche d'Images par le contenu", 2008.
- [6] M. Aritsugi, M. Tabata, H. Fukatsu, Y. Kanamori, et Y. Funyu. Manipulation of Image Objects and Their Versions under CORBA Environment. Dans 8th International Workshop on Database and Expert Systems Applications, DEXA '97, pages 8691, Toulouse, France, 1997. Roland Wagner, Helmut Thoma (Eds.), IEEE-CS Press.
- [7] S. Kawashima, M. Tabata, Y. Kanamori, et Y. Masunaga. Versioning Model of Image Objects for Easy Development of Image Database Applications. Dans 7th International Workshop on Database and Expert Systems Applications, DEXA '96, pages 194200, Zurich, Switzerland, September 9-10 1996. Roland Wagner, Helmut Thoma (Eds.), IEEE-CS Press, ISBN: 0-8186-7662-0.

- [8] H. Samet. The Quadtree and Related Hierarchical Structures. *Computing Surveys*,16(2):187260, 1984.
- [9] H. Samet. The Design and Analysis of Spatial Data Structures. Addison Wesley, 1989.
- [10] Maude Manouvrier, Thèse Pour l'obtention du titre de Spécialité Informatique, Paris IX-Dauphine UFR Sciences des Organisations Objets similaires de grande taille dans les bases de Données, pp.49-75, 2000.
- [11] P.VAN. OOSTROM. « Reactive Data Structures for Geographic Information systems », Oxford University Press, ISBN: 0-19,823320-5, 1993.
- [12] Souhila DJERROUD et Lynda ZAOUI, Segmentation d'une Base d'Images Hétérogène en des Groupes d'Images Homogènes par une Méthode Issue du Domaine du Data Minig, 5th International Conference: Sciences of Electronic, Technologies of Information and Telecommunications March 22-26, 2009 – TUNISIA , B.P. 1505 El-Mnaouer, Oran,31036, Algeria.
- [13] http://obligement.free.fr/articles/traitement_images_5.php
(Le traitement d'images numériques - quadtree et découpages itératifs)
- [14] A.N.C. Kang, R.C.T. Lee, Chin-Liang Chang, et Shi-Kuo Chang. Storage Reduction Through Minimal Spanning Trees and Spanning Forests. *IEEE Transactions on Computers*, c-26(5):425435, mai 1977.