dF.S.D ... N° D'ordre : .....

Université Saâd DAHLAB de Blida

Faculté des sciences

Département d'Informatique

Mémoire présenté par :

AISSOU Ayoub

Pour l'obtention du diplôme de Master

Domaine : Mathématique et Informatique

Filière : Informatique

Spécialité : Ingénierie des logiciels

Sujet :

Indexation et recherche d'images par similarité visuelle basée sur les graphes

Devant le jury composé par :

Mr CHRIF ZAHAR Amine : promoteur

Mr HADJ YAHIA : président de jury.

Mr DJILALI Nahal : Examinateur

Année universitaire : 2011-2012

DEDICACE

Je dédie ce travail

A mes très chers parents qui sont toute ma vie et tout ce que j'ai de plus cher au monde, en témoignage de ma reconnaissance infinie pour les nombreux sacrifices qu'ils n'ont cessé de déployer pour moi et dont je serais à jamais redevable.

Que dieu les garde et leur procure la santé et le bonheur.

A mes frères, et ma sœur nana et sans oublier Rachid et Hamza.

A toute ma famille sans exception.

A tous mes amis.

Remerciement

Remerciements

Je tiens à remercier en premier lieu Dieu le tout puissant qui m'a donné le courage et la patience et qui a éclairé mon chemin pour achever ce travail dans les meilleures conditions.

Plus particulièrement je remercie monsieur ZAHAR Cherif Amine pour la confiance qu'il m'a accordé, pour son encadrement continu avec patience et gentillesse, m'a fait profiter de sa grande expérience, ses remarques constructives, ses orientations, et ses conseils qui ont grandement contribué à améliorer la qualité de ce mémoire.

Je tiens également à remercier l'ensemble des enseignants du département de l'informatique qui ont contribué à ma formation.

Dédicaces spéciale à tous nos camarades de la promotion 2010/2011, plus particulièrement à mes amies. Puisse dieu les aider tous à atteindre tous leurs buts.

Je tiens à adresser mes vifs remerciements aux membres du jury pour s'intéresser au sujet et avoir accepté de lire ce mémoire.

A tous ceux et à toutes celles dont les noms n'apparaissent pas sur cette page, qu'ils demeurent convaincus, que je ne les ai point oubliés et qu'ils soient assurés de ma profonde gratitude.

Merci.

AISSOU Ayoub

Résume

L'utilisation de graphes est motivée par le double intérêt qu'apportent ces derniers pour modéliser tous les objets d'une forme donnée et toutes les relations inter objets nécessaires pour la reconnaissance. Un exemple typique utilisé dans ce mémoire est celui de la recherche d'images par le contenu(RIPC). La représentation des images par des graphes implique le recours à des algorithmes d'appariement de graphes afin de comparer et de détecter la similarité entre les images. Par ailleurs la recherche dans une base de données d'image nécessite une réorganisation préalable de la base afin de faciliter la recherche, ce qui nous conduit à faire appel à des techniques de classification des images représentées par des graphes.

Dans un premier temps, nous utilisons un algorithme de segmentation pour extraire les informations correspondants à l'image requête. L'ensemble des correspondances est extrait, évalué et finalement le graphe résultat correspondant à l'image requête est généré. Dans un deuxième temps, nous utilisons deux algorithmes pour comparer le graphe requête avec la base de graphes (appariement entre les graphes). L'un pour trouver une solution initiale et l'autre pour améliorer la solution trouvée.

Finalement, nous proposons un système de recherche d'images par le contenu utilisant les graphes pour représenter leur contenu et les deux algorithmes précédemment décrits. D'une manière générale, les résultats présentés dans ce mémoire montrent l'intérêt potentiel d'utiliser les graphes pour la représentation géographique des couleurs. Ces résultats semblent valider le choix judicieux des graphes comme une solution de remplacement aux structures de données classiques à savoir les vecteurs. De plus, on voit clairement à travers les résultats obtenus que les algorithmes, développés dans ce mémoire, pourront jouer un rôle primordial comme un outil de mesure de similarité dans un espace aussi complexe que les graphes.

ABSTRACT

The abstract

This thesis is part of the general structural pattern recognition. It is particularly interested in modeling forms by graphs. The use of graphs is motivated by the double advantage brought by them to model all objects of a given shape and all inter necessary objects for recognition. A typical example used in this thesis is the image search by content (RIPC). However, the techniques presented in this thesis have a wider field than the INCP. The representation of images by graphs involves the use of graph matching algorithms to compare and detect the similarity between images. Also search in a database image requires a prior reorganization of the base to facilitate research, which leads us to use techniques of image classification represented by graphs.

At first we use a segmentation algorithm to extract information relevant to the query image. All correspondence is extracted; assessed and finally generated the graph matches for the query image. In a second step, we use two algorithms to compare the query graph with the graph data base (matching between the graphs). One to find an initial solution and the other to improve the solution found

Finally, we propose a system for image retrieval by content using graphs to represent their contents and the two algorithms previously described. In general, the results presented in this thesis indicate the potential of using graphs to represent shapes. These results seem to validate the choice of graphs as an alternative to classical data structures namely the vectors. In addition, it is clear through the results that the algorithms developed in this thesis, may play an important role as a tool for similarity measure in an area as complex graphs.

Table de matière

Table de matière

Table de matière

Table de matière

Table de matière

Table de figures

List des illustrations, graphiques et tableaux

Table de figures

Introduction générale

1. Problématique

Ces dernières années ont vu le développement de logiciels de recherche d'images. Mais la plupart de ces logiciels utilisent les techniques de recherche classiques comme la recherche textuelle des images qui ne se basent pas sur le contenu visuel de l'image et ne donnent pas des bons résultats. Aussi essayons nous dans ce travail d'adopter une technique de recherche par contenu basé sur les graphes qui fait la recherche par les descripteurs d'image comme la couleur, la texture et les formes.

L'utilisation des graphes dans la recherche par contenu est un domaine très vaste et difficile car les graphes sont des outils de représentation très puissants et universels utiles dans divers domaines des sciences notamment le traitement d'images.

Cependant, bien que puissants, les graphes sont des structures qui occupent un espace mémoire appréciable ce qui conduit également à un temps important lors de la recherche de graphes similaires.

2. Objectif

Avec ce sujet, nos objectifs sont :

- Faire un état de l'art des méthodes et des systèmes d'indexation et recherche d'image par contenu existants.
- Faire un état de l'art de l'utilisation des graphes dans l'analyse des images.
- Construire un système d'indexation et recherche d'images par similarité visuelle basées sur les graphes. Le but est de trouver des caractéristiques permettant d'indexer l'image et de retrouver des images similaires lors d'une requête en fonction du contenu des images. En développant ce système, nous développons une interface qui permet à l'utilisateur de faire la requête et de prendre les résultats ainsi que les mesures qui permettent d'évaluer la pertinence de la recherche.
- De mesurer les difficultés inhérentes à l'utilisation des graphes et d'essayer d'apporter des solutions.

3. Organisation de mémoire

Afin d'atteindre les objectifs cité ci-dessus, notre mémoire s'articulera autour de deux parties :

Chapitre I : contient des concepts fondamentaux sur l'image et les systèmes de recherches d'image par contenu.

Chapitre II : contient un état de l'art sur les graphes.

Chapitre III : contient la conception d'un système d'indexation et recherche d'image par similarité basée graphes.

Chapitre IV : contient des tests de validation sur l'application ou bien des exemples et des interprétations de ces exemples.

Chapitre I :

Généralités sur les images

1. Introduction

Ces dernières années nous avons remarqué la croissance d'utilisation des images sur le web ce qui implique que l'image devient une chose très importante dans notre vie.

2. L'image

La définition du terme « image » lui-même, telle qu'elle est donnée par exemple par le Petit Robert, englobe une multitude de significations distinctes. Cela va de la « reproduction exacte ou représentation analogique d'un être, d'une chose », à la « représentation mentale d'origine sensible » ou à des concepts plus physiques comme un « ensemble des points » où vont converger des rayons lumineux (cas des images optiques).

Une image numérique est composée d'unités élémentaires (appelées pixels) qui représentent chacun une portion de l'image. Plus il y a de pixels dans une image, mieux cette dernière est définie, donc plus elle supportera un agrandissement de qualité.

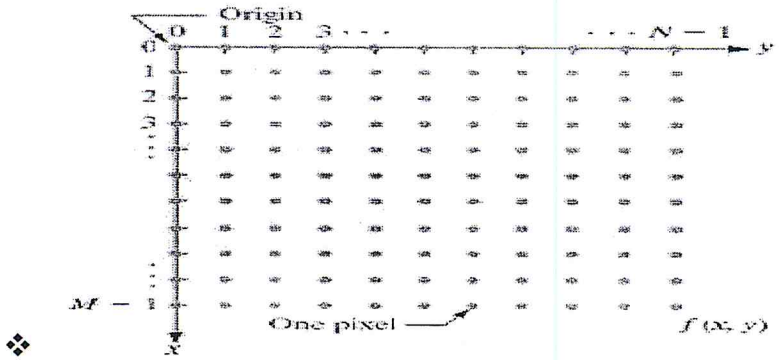

Figure I.1 : Représentation de l'image numérique. **[1]**

3. Les descripteurs d'image

Puisque l'on souhaite construire des systèmes de recherche d'images qui soient utilisables par l'être humain, et que c'est l'être humain qui donne un sens à ce qu'il voit, il peut être intéressant de s'inspirer du système de perception humain pour choisir les espaces visuels afin de s'approcher le plus possible de la compréhension qu'à l'être humain de l'image, et ainsi réduire le fossé sémantique.

3.1. La couleur

La couleur est le descripteur visuel le plus employé. Certainement car c'est le plus perceptible, Elle est définie par sa longueur d'onde, ou par un mélange de langueur de langueurs d'ondes. La science qui s'occupe le domaine des couleurs est la colorimétrie. Différentes définitions ont été données par d'auteurs, parmi lesquels :

- **[1]** définit la colorimétrie comme une science qui tente de qualifier une source lumineuse de manière absolue.
- **[2]** considère la colorimétrie comme la standardisation du triplet lumière-objet-observateur. La compagnie internationale de l'éclairage CIE a défini plusieurs espaces pour la représentation de la couleur. Ces espaces sont différenciés et classés selon l'uniformité de perception.

L'uniformité de perception dans un espace de représentation est vérifiée par les deux critères suivants :

- La distance d (c1, c2) entre les deux couleurs c1 et c2 est correct, si et seulement si, la valeur de cette distance se rapproche de la différence perçue par l'œil humain.
- La distance d (ci, c1)=n*d (ci, c2) est correct, si et seulement si, l'œil humain perçoit la couleur c1 n-fois plus éloignée de la couleur ci que la couleur c2.La distance utilisée est la distance euclidienne.

Les différences espaces de représentation de la couleur sont :

a. L'espace RGB

En 1931, la CIE a défini l'espace RGB **[3]** comme suit: le R:red, le G:green, B:blue. Cet espace est donc représenté par les trois primaires monochromatiques de couleurs, dont les longueurs d'ondes sont respectivement 700.0 nm, 546.1 nm, 435.8 nm, et par le blanc de référence qui n'est pas unique.

L'inconvénient majeur de cet espace est l'impossibilité de représenter toutes les couleurs par la superposition des trois spectres, à cause de l'existence d'une partie négative, qui peut poser des problèmes si on travaille en synthèse additive. Cet espace ne permet pas de reconstitue vraiment toutes les couleurs perceptibles par l'œil humain **[4]**.

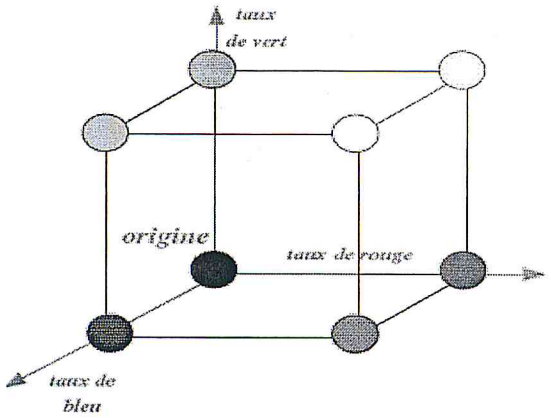

Figure I.2 : représentation de l'espace RGB. [4]

b. L'espace XYZ

L'espace XYZ est défini par trois primaires X, Y et Z dites virtuelles. Le passage à l'espace XYZ est effectué par une transformation linéaire sur l'espace RGB. La figure suivante présente le diagramme de chromaticité **xy** qui permet de tenir compte de la sensibilité de l'œil humaine.

Cet espace n'est pas perpétuellement uniforme **[5]**, car la distance entre deux couleurs du diagramme de chromaticité ***xy*** n'est pas perçue de la même façon que le système visuel humaine (SVH).

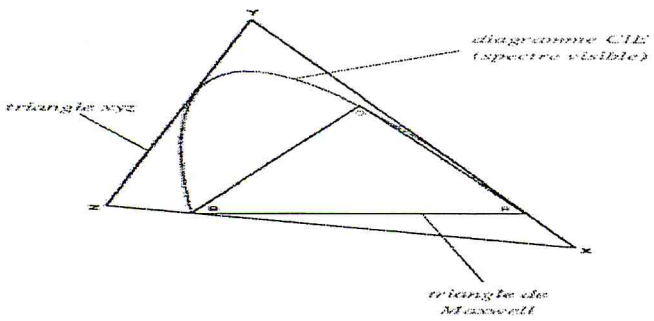

Figure I.3 : représentation de l'espace XYZ. [5]

c. L'espace L*u*v

En 1976, L'espace L*u*v devient un standard de la CIE **[6]**, où la luminosité L représente la clarté, u l'opposition de couleurs vert-rouge et v l'opposition de couleurs bleu-jaune. L'espace L*u*v est un espace perpétuellement uniforme.

d. L'espace L*a*b

L'espace L*a*b est un espace perpétuellement uniforme introduit par la CIE, en 1976, où la luminosité représente la clarté, et a*, b* représentent respectivement l'opposition de couleurs vert-rouge et l'opposition de couleurs bleu-jaune. Les axes a* et b* sont orientés (vert pour a*<0 et b*=0, rouge pour a*>0 et b*=0, etc.).

Cet espace est obtenu par des relations non-linéaires à partir de l'espace XYZ. A cause de ces relations non-linéaires, il n'est pas possible de définir un diagramme de chromaticité.

e. L'espace L*C*h

L'espace L*C*h désigne la luminance L*, la chroma C* et la teinte h*. Cet espace est le résultat du passage des espaces L*a*b et L*u*v en coordonnées semi-polaires.

f. L'espace HSV (Hue-Saturation-Value)

Cet espace s'appelle aussi le cône hexagonal. Il représente la couleur selon trois entités: la teinte (H), la saturation (S) et l'intensité ou luminosité (V).La figure suivante présente le cône hexagonal de l'espace HSV.L'espace HSV simule le comportement visuel humain. Cet espace est perceptuel mais pas uniforme.

Il existe d'autres espaces de représentation de couleur comme : l'espace CMY qui est dédié à l'impression couleur, l'espace d'Ohta … etc.

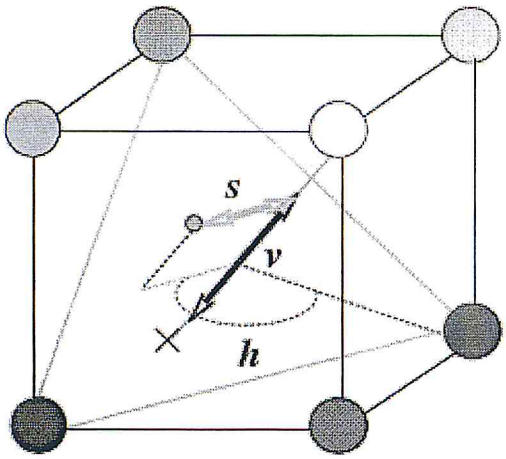

Figure I.4 : représentation de l'espace HSV. [6]

3.2. La texture

Il n'existe pas de définition pertinente de la texture. Cependant, une définition de sens commun est la suivante : la texture est la répétition d'éléments de base construits à partir de pixels qui respectent un certain ordre. On peut distinguer deux types extrêmes de textures, entre lesquels se positionnent toutes les textures :

- la première est déterministe(ou régulière) et défini référence à une répétition spatiale d'un motif de base dans différentes directions.

Cette approche structurelle correspond à une vision macroscopique des textures. La peau des reptiles est un exemple de texture macroscopique dans lequel la primitive est l'écaille.

- La seconde est probabiliste(ou aléatoire) et cherche à caractériser l'aspect chaotique qui ne comprend ni motif localisable, ni fréquence de répétition principale. Cette approche est macroscopique et l'herbe en est un excellent exemple.

L'analyse de texture se fait par la matrice de cooccurrence de Haralick ou la matrice de langueur de plage, qui nous permet d'obtenir des informations sur la texture macroscopique. Pour la texture microscopique, des calcules statistiques au niveau de gris sont utilisés. Le calcule des attributs de textures mixte s'effectue par l'analyse de Fourier et l'analyse en ondelettes.

La texture peut nous aider dans l'aspect sémantique, car elle différencie les parties de l'image dont les descripteurs de couleurs sont identiques. Par exemple la mer (texture des vagues).

a. La matrice de cooccurrence

La texture d'une image peut être interprétée comme la régularité d'apparition de couple de niveau de gris selon une distance donnée dans l'image. La matrice de cooccurrence contient les fréquences relatives d'apparition des niveaux de gris selon quatre directions : $\theta=0$, $\theta=\pi/2$, $\theta=\pi/4$, $\theta=3\pi/4$. La matrice de cooccurrence est une matrice carrée n*n où n est le nombre de niveau de gris de l'image.

Une fois la matrice de cooccurrence calculé, différentes attributs de texture peuvent être extraire comme : l'homogénéité, l'entropie, le contraste, l'inertie, la variance, et la corrélation.

b. La transformée de Fourier

La transformée de Fourier est défini comme suit:

$$F(u, v)=\iint I(x, y)\, e^{-i(ux+vy)}\, dxdy$$

I (i, j) : valeur d'un pixel de l'image.
F (u, v) : le coefficient de la transformée de l'image.

La transformée de Fourier permet de transformer les cordonnées de l'image du domaine spatial au cordonnées du domaine fréquentiel de l'image. Le module de transformation de Fourier ou spectre de Fourier est souvent utilisé pour éviter l'interprétation de la phase quand on travaille dans des plans complexes.

Les avantages de la transformée de Fourrier est qu'il y a des invariances. La phase est invariante à la luminosité, elle est stable jusqu'à un changement d'échelle de20%. L'amplitude est aussi invariante à la translation. Normalement, on calcule les signatures des images en utilisant des coefficients de la transformée de Fourrier discrète ou la transformée de Fourrier rapide **[7].**

c. Les ondelettes

La transformée en ondelettes est un outil de mathématique récent qui décompose un signal en fréquences en conservant une localisation spatiale. Le signal de départ est projeté sur un ensemble de fonctions de base qui varient en fréquences et en espace. Ces fonctions de base s'adaptent aux fréquences du signal à analyser. Cette transformation permet d'avoir une localisation en temps et en fréquences du signal analysé.

3.3. La forme

Les attributs de la forme sont utilisés pour caractériser les objets dans l'image. Ils ne sont pas utilisés qu'avec une image segmentée. La forme contient deux catégories: les descripteurs basés région et les descripteurs basés frontière.

a. Les descripteurs basés région

Le but de ce descripteur est de caractériser l'intégralité de la forme d'une région à travers les moments invariants.

- **Les attributs géométriques de région**

Il existe différents types de forme que peuvent prendre les objets d'une scène, les attributs géométriques de région permettent les distingués. A travers la segmentation de l'image en différentes régions, des calculs peuvent se faire sur les régions comme: la surface, le centre de masse, la langueur de contour, la compacité …

Le but de ces attributs est de distinguer les différents types de forme que peuvent prendre les objets d'une scène, une segmentation en région de l'image est nécessaire pour pouvoir calculer sur les différents régions de l'image les données suivantes: la surface, le centre de masse, la langueur de contour, la compacité …

- **les moments géométriques**

Le but des moments géométriques est de décrire une forme à l'aide de propriétés statistiques, HU a introduit sept invariants aux translations, rotation et changement d'échelle, appelée moment de HU. Les moments géométriques consomment du temps de calcul, mais il son simple à manipuler.

b. Les descripteurs basés frontière

Dans cette catégorie, on utilise les descripteurs de Fourier afin de caractériser les contours d'une forme.

4. Conclusion

Nous avons vu dans ce chapitre les concepts fondamentaux ou bien des généralités sur l'image ainsi que les descripteurs d'image. Et dans l'étape suivante on va détailler et expliquer ou bien éclairer tous les choses qui concernent les systèmes de recherche d'images par contenu.

PARTIE II : LES SYSTEMESD'INDEXATION ET RECHERCHE D'IMAGES PAR CONTENU

1. Introduction

L'expression « recherche d'images par le contenu » (« Content-Based Image Retrieval, CBIR, en Anglais) remonte aux travaux de Kato en 1992. Son système, ART MUSEUM, permet de retrouver des images d'art par couleurs et contours. Le terme s'est étendu par la suite à tout procédé permettant de rechercher des images selon des descripteurs, pouvant être de type « signal », comme la couleur et la forme, mais également symboliques. Comme le remarquent les auteurs d'un rapport important sur les systèmes de recherche par le contenu **[8]**, retrouver des images indexées manuellement par des mots clefs n'est pas de la recherche par le contenu au sens où le terme est généralement compris, même si ces mots clefs décrivent le contenu effectif de l'image.

Les **applications** des systèmes de recherche d'images existants (et donc les collections d'images) sont variées. Elles incluent des applications judiciaires : les services de police possèdent de grandes collections d'indices visuels (visages, empreintes) exploitables par des systèmes de recherche d'images.

Les applications militaires, bien que peu connues du grand public, sont sans doute les plus développées **[8]** : reconnaissance d'engins ennemis *via* images radars, systèmes de guidage, identification de cibles *via* images satellites en sont des exemples connus. Le journalisme et la publicité sont également d'excellentes applications. Les agences de journalisme ou de publicité maintiennent en effet de grosses bases d'images afin d'illustrer leurs articles ou supports publicitaires. Cette communauté rassemble le plus grand nombre d'utilisateurs de recherche par le contenu (davantage pour les vidéos) mais l'aide apportée par ces systèmes n'est absolument pas à la hauteur des espoirs initiaux (**[8]**).

D'autres applications incluent: le diagnostic médical, les systèmes d'information géographique, la gestion d'œuvres d'art, les moteurs de recherche d'images sur Internet et la gestion de photos personnelles. C'est sur ce dernier thème que nous concentrons notre travail. Nous focalisons donc notre étude de l'état de l'art sur les approches dont la cible (utilisateurs, contenu des images) est peu spécialisée. Ces approches, devant nécessairement faire face aux problèmes dont nous avons parlé en introduction (problématique du temps de recherche dans des bases de grandes dimensions et pertinence de la recherche par le contenu), sont par conséquent les plus pertinentes dans notre cadre.

Concevoir un système permettant d'assister des utilisateurs dans leurs tâches de recherche d'images pose des **problèmes** variés. Dans **[8]** les difficultés suivantes sont identifiées :

- Comprendre les utilisateurs d'images et leurs comportements : de quoi les utilisateurs ont-ils besoin ?
- Identifier une manière « convenable » de décrire le contenu d'une image.

  C'est une tâche rendue difficile par la subjectivité intrinsèque aux images.

❖ Extraire des « descripteurs » des images brutes.
❖ Pouvoir stocker de manière compacte un grand nombre d'images.
❖ Comparer requêtes et images stockées de manière à refléter les jugements de similarité humains.
❖ Accéder efficacement aux images par leur contenu
❖ Fournir des interfaces utilisables

A cela, ajoutons la difficulté majeure, dont nous avons parlé dans l'introduction, connue sous le nom de « **Malédiction de la dimension**» **[9]**.

Il convient donc d'abord de montrer quelles sont les approches existantes ainsi que leurs limitations. Nous commençons par rappeler les composants d'un système de recherche d'images par le contenu, puis nous présentons une taxonomie des systèmes selon leur niveau d'abstraction en donnant systématiquement des exemples.

2. Composants d'un CBIR

Nous décrivons brièvement ici les caractéristiques communes à la plupart des approches : le traitement de la base d'images, les requêtes puis la mise en correspondance et la présentation des résultats. La Figure 1 illustre l'ordonnancement de ces étapes :

Dans un premier temps (2), des descripteurs sont calculés à partir de chaque image de la collection (1), ils peuvent être de type signal ou/et symbolique (le vocabulaire d'indexation).Les données extraites (à présent représentatives du contenu de l'image du point de vue du système) constituent la base d'index (3). Les requêtes de l'utilisateur (4) sont alors transformées afin d'être comparables avec la base d'index (5) ; une mise en correspondance (6) entre la requête transformée et la base d'index permet ensuite de produire le résultat de la requête (7). Il se peut également que le système possède des composantes liées à la

personnalisation, comme par exemple l'extraction, le stockage et l'utilisation d'un profil d'utilisateur.

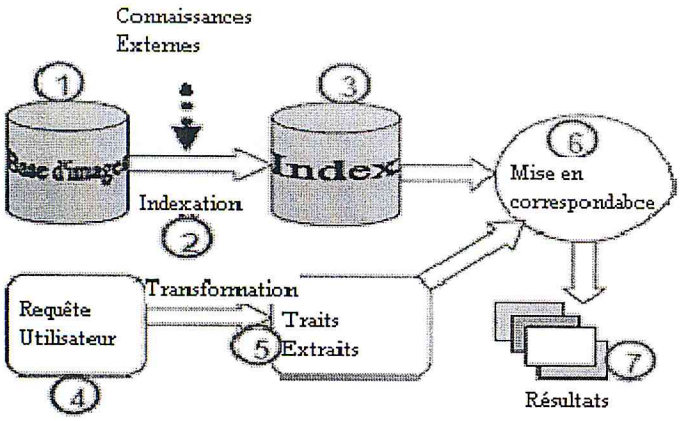

Figure I.5 : Systèmes d'indexation et recherches d'images. **[10]**

a. La base d'images

La collection (ou base) d'images est la donnée principale du système. Les bases d'images varient d'abord par leur **taille** : la majorité des systèmes est conçue pour des bases de quelques centaines ou milliers d'images **([10]).** Ce nombre peut cependant s'approcher du milliard (880 millions d'images) lorsque la base est constituée par les images collectées par des robots sur Internet. La taille de la base d'image impose des contraintes sur la complexité des traitements effectués sur chaque image. Il en résulte que la comparaison qualitative entre des systèmes travaillant sur des bases de tailles très différentes est peu pertinente.

Le **type d'image** composant la base varie également : des portraits en noir et blanc, des peintures chinoises anciennes, des images personnelles, des images de tissus humains, etc. Le type d'image influe fortement sur la conception globale du système, particulièrement sur les descripteurs de bas niveau calculés. D'une manière générale, plus la variabilité intra et inter images est importante, plus le système doit être riche et précis (et plus le problème d'indexer/rechercher ces images est difficile). **[11]**

Finalement, les collections diffèrent par leur **stabilité**, c'est-à-dire le taux de changements (ajouts d'images, retraits, etc.) en fonction du temps. Faible pour une collection d'images représentant les œuvres d'un peintre ne créant plus, elle peut être très forte lorsque, par exemple, on s'intéresse aux images de la Toile ou à l'actualité.

b. L'indexation

L'indexation est l'ensemble des processus aboutissant à la construction d'un index de l'image. Contrairement à d'autres types de données, comme le texte, il n'est pas possible d'utiliser les images *directement* dans un CBIR. Il faut caractériser les images par des informations à la fois discriminantes et invariables à certains paramètres (comme la taille de l'image, l'angle de la prise de vue, etc.). L'indexation peut être **fixe** : les descripteurs calculés sont toujours les mêmes. L'indexation peut aussi être **évolutive** : les descripteurs s'adaptent à l'utilisateur ou au contexte dans le temps, ce qui permet de renforcer l'adéquation système/utilisateur. **[12]**

L'indexation peut être **générique** (indexation de photographies diverses dans **[12]**), pouvant caractériser des collections hétérogènes, ou **spécifique** (indexation de peintures chinoises dans **[12]**), adaptée à un type d'image particulier. Une collection hétérogène est par exemple constituée de photographies personnelles, mettant en scène diverses entités physiques dans des conditions de prise de vue variables. Indexer une telle collection impose l'usage de descripteurs suffisamment génériques (la couleur par exemple), c'est-à-dire qui caractérisent une propriété discriminante applicable à la plupart des entités physiques. A l'inverse, indexer une collection d'images très spécifiques (des empreintes digitales par exemple) requiert l'utilisation de descripteurs également très spécifiques qui, par ailleurs, ne conviendraient probablement pas à une collection hétérogène.

La phase d'indexation peut inclure une étape de **segmentation**, afin de caractériser des régions homogènes de l'image (**[13]**) ou bien indexer l'image dans sa globalité **[13].** La segmentation de l'image précède généralement l'indexation individuelle des régions de l'image et cela permet, outre le fait d'accéder à des parties de l'image, de calculer des descripteurs de forme.

Enfin, l'indexation varie d'un système à l'autre par son **niveau d'abstraction** : extraire des histogrammes de couleurs est une opération directe, alors que reconnaître des personnes ou des objets est beaucoup plus complexe et requiert un apprentissage préalable.

c. La gestion des index

Elle concerne la manière dont sont gérés les index des images : stockage et accès. La gestion des index, anecdotique pour une collection de taille modeste, devient une

préoccupation essentielle lorsque l'on travaille sur une base de taille conséquente. La manière la plus basique de stocker les index est la liste séquentielle, que ce soit en mémoire ou dans un fichier. Cependant, lorsque le nombre d'images augmente, le temps d'accès à une image augmente linéairement et il est souvent nécessaire d'organiser les index de manière hiérarchique, sous forme d'arbres (organisés selon les descripteurs), ou de tables de « hash code » par exemple, afin d'accélérer l'accès à l'information.

d. Les requêtes

Le type de requête proposé découle de choix fait en amont, au niveau de l'indexation. Dans des systèmes où seuls des descripteurs de bas niveau sont extraits, les requêtes ne peuvent être que de bas niveau : requête par « image exemple », par croquis ou par manipulation directe des traits de bas niveau. Dans ces systèmes, des descripteurs sont extraits à partir de la requête (une image, un croquis…) et sont comparés aux descripteurs calculés à partir des images de la base (les index des images).

A l'opposé, dans des systèmes proposant plus d'abstraction (**[14]**), les requêtes peuvent être sémantiques (textuelles par exemple). Par exemple dans **[14]**, les images sont indexées par des « catégories sémantiques visuelles », ce qui permet à un utilisateur de formuler des requêtes sémantiques (« Je veux des images prises à l'extérieur. »).

e. Analyse de la requête

Cette étape a pour but de transformer la requête utilisateur pour la rendre comparable avec les index de la base d'images ; elle consiste donc généralement à extraire les mêmes types de descripteurs que ceux extraits de la base d'image lors de l'indexation.

f. Mise en correspondance requête / base

Il s'agit d'estimer dans quelle mesure une image (son index) satisfait une requête donnée. Dans le contexte de la recherche d'images, cela se ramène souvent à calculer la similarité entre les caractéristiques extraites de la requête et les caractéristiques de chaque image dans la base. Cela aboutit généralement à une valeur de correspondance qui caractérise la pertinence (du point de vue du système) d'une image par rapport à la requête. Cette mise en correspondance peut être simple (comparaison d'histogrammes) ou complexe (comme dans par exemple, avec une mise en correspondance qui tient compte de l'arrangement spatial des régions).

La phase de mise en correspondance peut également inclure une pondération des descripteurs (où chaque descripteur est pondéré par rapport à son pouvoir discriminant dans la base). Pondérer les descripteurs permet d'éliminer une partie du bruit dans la mesure où les descripteurs les moins pertinents voient leur influence diminuer dans l'évaluation de la similarité requête/image.

La mise en correspondance peut également inclure un bouclage de pertinence. Le but est également d'éliminer le bruit (augmenter la précision) en tentant de converger vers une précision maximale.

g. La présentation des résultats

Dans la grande majorité des systèmes disponibles **[15]**, le résultat d'une requête est présenté sous la forme d'une liste d'images (réduites à des vignettes) ordonnées par pertinence décroissante. Parfois cette présentation prend d'autres formes, comme par exemple l'œil de poisson (FishEye View) **[16].** L'avantage des images par rapport aux documents textuels est qu'il est possible de visionner d'un coup d'œil l'intégralité du document, ce qui permet de visualiser un grand nombre de résultats et de les comparer plus rapidement. Comme indiqué plus haut, la présentation des résultats est souvent couplée avec une possibilité d'interaction, qui permet par exemple de raffiner une requête en indiquant au système les résultats pertinents et ceux qui ne le sont pas (bouclage de pertinence), et de permettre ainsi une reformulation automatique de la requête.

3. Représentation des images dans un CBIR

Dans la majeure partie des systèmes existant, les images sont représentées avec des descripteurs de bas niveau, i.e., en termes de couleur, texture, formes, points d'intérêt ou par des descripteurs hybrides.

4. Conclusion

Dans cette rechercher bibliographie sur l'indexation et la recherche d'images par contenu, nous avons exploré, dans un premier lieu des généralités sur l'image avec les différents descripteurs de l'image. En seconde lieu, nous avons dressé quelconques systèmes existe à l'heure actuelle.

Les systèmes de recherche d'image par contenu permettent aux utilisateurs d'accéder à l'information visuelle de manière plus adaptés que la manière classique ou traditionnelle.

ChapitreII :

L'état de l'art sur les graphes

1. Introduction

La représentation sous forme de graphe est utilisée dans différentes applications d'analyse d'images. Dans ces applications, l'image est traitée pour produire un graphe représentant les composants et les relations entre eux. Cette représentation structurelle puissante a prouvé sa flexibilité en traitement d'une grande variété de types d'images (les anciens documents, les plans électriques et architecturaux, les images naturelles, les images médicales ...). Les travaux, qui ont fait recours aux graphes, ont visé une représentation qui préserve l'information topographique de l'image ainsi que les relations entre les composantes.

Actuellement, plusieurs approches dérivées des graphes pour l'analyse d'images ont été proposées notamment pour les systèmes de recherche par contenu, la segmentation et l'indexation.

2. Les graphes

Un graphe est défini par un couple $G = (V, E)$ où V est un ensemble fini de sommets et E est un ensemble fini d'arêtes.

3. Représentation des graphes en machine

Il existe deux types de représentation d'un graphe en machine : une représentation matricielle et une représentation par listes chaînées. Dans le cas de la représentation matricielle, il y a deux façons de procéder : représentation par une matrice sommets-arcs ou par une matrice sommets-sommets.

a. Matrice d'incidence sommets-arcs

On définit la matrice $A = (a_{ij})$ comme suit :

$$a_{ij} = \begin{cases} 1 \text{ si } i \text{ est l'extrémité initiale de } u_j \\ -1 \text{ si } i \text{ est l'ectrémité terminale de } u_j \\ 0 \text{ si } i \text{ n'est pas une extrémité de } u_j \end{cases}$$

$$A = \begin{matrix} 1 & -1 & -1 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{matrix}$$

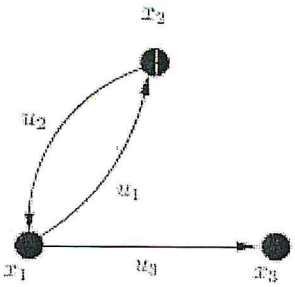

Figure II.1 : représentation du graphe par matrice d'incidence sommets-arcs.[17]

Sur chaque colonne, il y a un seuil et un seul -1 sauf dans le cas d'une boucle où il n'y a, par convention, qu'un seul 1 (c'est le cas de la dernière colonne : boucle *U6*). Dans le cas des graphes non-orientes, on obtiendrait pratiquement la même matrice que précédemment : il suffit de remplacer les -1 par des 1. La place mémoire d'une telle matrice est de $\boldsymbol{n \times m}$. De plus, si le graphe est value (valeur sur chaque arc) il est nécessaire de stocker cette valuation dans un tableau de dimension m. Si le nombre d'arcs m est plus important que le nombre de sommets n, il est préférable de représenter un graphe par une matrice d'adjacence sommets-sommets.

b. Matrice d'incidence sommets-sommets

On définit généralement la matrice $\boldsymbol{A = (a_{ij})}$ comme suit :

$$\boldsymbol{a_{ij}} = \begin{cases} \boldsymbol{1\ si\ (i,j) \in U} \\ \boldsymbol{0 \quad\quad si\ non} \end{cases}$$

$$A = \begin{matrix} \mathbf{0} & \mathbf{1} & \mathbf{1} \\ \mathbf{1} & \mathbf{0} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{matrix}$$

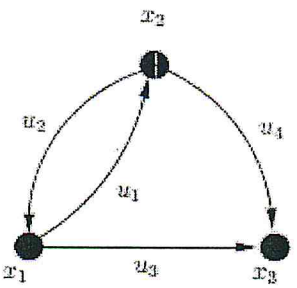

Figure II.2 : représentation du graphe par matrice d'incidence sommets-sommets. [17]

Dans le cas des graphes non-orientés, il suffit d'ajouter les 1 qu'il faut de façon à rendre la matrice symétrique. La place mémoire d'une telle matrice est de$\boldsymbol{n^2}$. Dans le cas d'un graphe valué, il est nécessaire de disposer d'une autre matrice carrée de taille $\boldsymbol{n^2}$ pour stocker les valeurs. Lorsque le nombre de sommets est faible par rapport à$\boldsymbol{n^2}$, il y a beaucoup de 0 dans

la matrice, il devient alors plus intéressant de représenter les graphes sous forme de listes d'adjacence.

c. Représentation par listes d'adjacence

On s'aperçoit que les matrices précédentes possèdent beaucoup de 0. La représentation par listes d'adjacence évite le stockage de tous ces 0. Dans cette représentation, le graphe est stocké sous forme de listes chainées de sommets. Chaque sommet pointe sur la liste de ses successeurs (graphe orienté) ou voisins (graphe non-orienté).La représentation est la suivante :

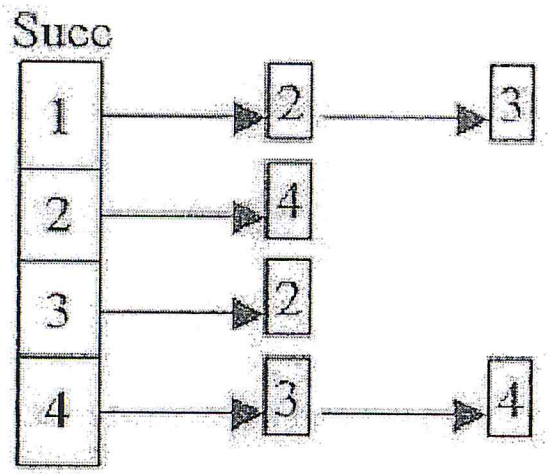

Figure II. 3 : Représentation de graphe par liste d'adjacence. **[17]**

La place mémoire de cette représentation est $\boldsymbol{n+m}$ où n est le nombre de sommets et m est le nombre d'arcs. Pour un graphe orienté et $\boldsymbol{n+2m}$ pour un graphe non orienté. Si le graphe est valué, il faut insérer la valuation de l'arc (i,j) dans la liste des successeurs de i, avec le sommet j.

4. Représentation des images par des graphes

Généralement avant tout traitement les images sont segmentées en régions. Chaque sommet du graphe représente alors une région de l'image et les arcs représentent les différentes relations possibles entre les régions (connectivité, distance...). L'intérêt principal de l'utilisation des graphes pour la représentation d'images est l'intégration d'informations spatiales dans la modélisation. En effet les représentations classiques telles que les histogrammes de couleurs, les descripteurs de textures, etc... ne donnent aucune information sur la façon dont les régions d'intérêt de l'image sont agencées. Au contraire, la représentation

par les graphes permet de décrire la structure de l'image comme la façon dont les régions sont disposées les une par rapport aux autres. En outre, selon les types de relations choisies, cette représentation en graphe peut être invariante à certaines transformations telles que les rotations de l'image ou les translations de certaines parties de l'image **[36].**

5. Appariement des graphes

Afin de comparer deux graphes, il est nécessaire de mettre en correspondance leurs sommets, c.-à-d de définir un appariement (matching) entre ces graphes.

6. Les types d'appariement : il y'a plusieurs types de similarité. Dans la partie suivante on présente les types les plus connus :

- **Appariement bijectif** : cardinalité(1,1).
  Tous les sommets d'un graphe sont appariés à un et un seul sommet de l'autre graphe et inversement. Il n'existe pas donc deux sommets d'un graphe liés à un même sommet de l'autre graphe.
- **Appariement injectif** : cardinalité= (1,0..1)
  Un appariement injectif de *G1* à *G2* lie chaque sommet du graphe *G1* à au maximum un sommet du graphe *G2*. Il est impossible d'apparier deux sommets de *G1* à un même sommet de *G2*.
- **Appariement univoque** : cardinalité = (0..1,0..1)
  Dans un appariement univoque entre deux graphes G1 et G2, chaque sommet d'un graphe est apparié au maximum à un sommet de l'autre graphe. Chaque sommet n'a donc que deux possibilités : soit il n'est pas apparié, soit il est apparié à un sommet unique.
- **Appariement multivoque** : cardinalité = (0..|V'|,0..|.V|)
  Chaque sommet d'un graphe est apparié à un ensemble (éventuellement vide) de sommets de l'autre graphe. Dans un appariement multivoque, un sommet est peut-être apparié en même temps à plusieurs sommets

7. Mesures de similarité

Mesurer la similarité d'images est un problème qui se pose dans de nombreuses applications, que ce soit pour rechercher des images, les classifier ou encore les confronter à des modèles pour les "reconnaître". Quand les images sont représentées par des graphes, on peut mesurer la similarité de deux images en appariant les sommets des graphes associés aux

images de façon à retrouver leurs caractéristiques communes. Différentes sortes d'appariements de graphes, impliquant différentes mesures de similarité, ont été proposées, e.g. l'isomorphisme de (sous-)graphes pour mesurer l'équivalence ou l'inclusion de deux graphes, et la distance d'édition de graphes pour mesurer le coût de transformation d'un graphe en un autre. Ces différents appariements sont "univoques" dans le sens où un sommet d'un graphe ne peut être apparié qu'à au plus un sommet de l'autre graphe, Dans les mesures des graphes basées sur un appariement multivoque des sommets des graphes, chaque sommet peut être apparié à un ensemble de sommets de l'autre graphe. Cet aspect nous permet notamment de comparer des graphes décrivant des images à différents niveaux de granularité, et de prendre en compte les problèmes de sur- et de sous-segmentation des images. Dans la suite nous détaillerons les mesures les plus connues dans la littérature.

7.1. La distance d'édition

La distance d'édition pour les graphes est l'extension de la distance d'édition pour les chaînes des caractères (ou distance de Levenshtein) aux graphes **[18].** La distance d'édition est une mesure très commune de similitude pour les graphes et des variantes ont été proposées avec succès dans beaucoup de domaines d'application telle que par exemple la reconnaissance des visages. La distance d'édition entre deux graphes est le nombre minimum des opérations d'édition qui sont nécessaires pour transformer un graphe en un autre. Les principales opérations d'édition sont, la suppression ou l'insertion des nœuds ou arcs, et le changement des attributs des nœuds ou des arcs.

Pour calculer la distance entre deux graphes Robles-Kelly **[19]** propose une méthode de conversion d'un graphe à une séquence de chaînes de caractères afin d'appliquer des techniques de mesure de la distance d'édition de chaîne de caractères. La problématique principale de cette mesure est la détermination du coût minimal des opérations d'éditions effectuées **[19].**

7.2. Isomorphisme de graphes

Un graphe G est isomorphe à un Graphe G' si on trouve un appariement univoque permettant de retrouver tous les sommets et tous les arcs des deux graphes.

7.3. Isomorphe de sous-graphe partiel

Un graphe G est un sous-graphe partiel de G' si on trouve un appariement univoque des sommets de G aux sommets de G' permettant de retrouver toutes les caractéristiques de G.

7.4. Isomorphe de sous-graphe

Le problème de l'isomorphisme de sous-graphes est un cas particulier du problème de l'isomorphisme de sous-graphes partiel où la condition suivante est ajoutée : tous les couples de sommets de G qui ne sont pas reliés par un arc doivent être appariés à des sommets de G' qui ne sont également pas reliés par un arc.

7.5. Plus grand sous-graphe partiel commun

De deux graphes G et G' est le plus grand graphe (par rapport au nombre de sommets et d'arcs) qui soit isomorphe à des sous-graphes partiels de G et de G'.

7.6. Le plus grand sous-graphe commun

Cette méthode de mesure de similarité entre deux graphes proposée par Bunke et Shearer dans **[40]** est basée sur le plus grand sous-graphe commun de deux graphes. Un graphe G est appelé un sous-graphe commun aux deux graphes G1 et G2, s'il est un sous graphe de G1 et G2. Un sous-graphe commun G de deux graphes G1 et G2 est dit maximal (le plus grand) s'il n'existe aucun autre sous-graphe commun de G1 et G2 avec un ordre (taille, en terme de nombre de sommets) plus grand que celui de G. Dans [41], le plus grand sous-graphe commun est noté « ***mcs*** » (***maximal common su bgraph***). La mesure de similarité entre deux graphes non-vides G1 et G2 est définit par :

$$\boldsymbol{d_{mcs}(G1,G2) = 1 - \frac{|mcs(G1,G2)|}{max\{|G1|,|G2|\}}}$$

Où |***mcs*** (G1, G2)| est le nombre des sommets du ***mcs*** et | G1 | et | G2| sont respectivement le nombre de sommets de G1 et G2. Cette mesure selon [42] respecte les propriétés suivantes :

$$\boldsymbol{0 \leq d_{mcs}(G1,G2) \leq 1}$$

$$\boldsymbol{d_{mcs}(G1,G2) = 0} \leftrightarrow \textit{G1 et G2 sont isomorpes l'un de l'autre}$$

$$\boldsymbol{d_{mcs}(G1,G2) = d_{mcs}(G2,G1)}$$

$$\boldsymbol{d_{mcs}(G1,G3) = d_{mcs}(G1,G2) + d_{mcs}(G1,G23)}$$

7.7. La notion de graphe médian

Le médian peut être un concept très utile pour avoir une représentation qui accumule une information globale de l'ensemble. Dans le domaine de l'analyse d'image par approches structurelles, le graphe médian, introduit par Jiang et Bunke, a pour objectif l'extraction de l'information essentielle à partir d'un ensemble de graphes dans un seul prototype (i.e. modèle).

Soit S un ensemble de graphes, le médian est définit comme le graphe dérivé de S qui a la plus petite somme de distances à tout les graphes dans l'ensemble S. Si la condition que le médian appartient à l'ensemble des graphes est requise, le médian de l'ensemble est obtenu. Sinon, un graphe médian généralisant est obtenu.

7.8. La mesure de Papadopoulos et Manolopoulos

Papadopoulos et Manolopoulos **[20]** présentent une mesure de similarité pour les graphes qui est aussi basée sur le concept des opérations d'édition. Ils proposent trois opérations primitives différentes, qui sont l'insertion de nœuds, la suppression de nœuds et la mise à jour de nœuds. Pendant que les opérations de suppression et d'insertion ont des significations évidentes, l'opération de mise à jour est nécessaire pour insérer ou supprimer les arcs incidents pour un nœud. En plus, Papadopoulos et Manolopoulos introduisent la séquence de degrés d'un graphe qui est la donnée en ordre décroissante des degrés de chaque nœud du graphe. La distance de similarité entre deux graphes est définit comme le nombre minimum des opérations primitives qui sont nécessaires pour que les deux graphes possèdent la même séquence de degrés.

8. Méthodes utilisant des graphes

Dans le domaine de l'analyse d'images, les graphes ont été largement utilisés dans différents sous-domaines. Dans notre travail, nous focalisons sur les exploitations des graphes les plus intéressants dans le domaine de la segmentation, la recherche par contenu.

8 .1 Segmentation

La segmentation est un processus qui consiste de découper une image en régions connexes présentent une homogénéité selon un critère qui permet de contrôler l'aspect final de la segmentation comme la couleur, la texture, la taille ...le regroupement de ces régions doit redonner limage initiale.

La segmentation à pour objectif de différenciée les zones d'intérêts (objets, fond,…), et l'extraction des informations qualitatives de l'image. Elle fournit une description de haut niveau : chaque région est connectée à ses voisins dans un graphe et chaque région porte une étiquette donnant des informations qualitatives comme sa taille, sa couleur, sa forme, son orientation. Les arcs de graphes peuvent être valués précisant si les deux régions sont connectées.

On peut regrouper les algorithmes de segmentation en trois grandes catégories :

8 .1.1 Approches GLOBALES

Est une segmentation basée sur les pixels, par exemple : seuillage, histogramme (segmentation par classification monodimensionnelle).L'idée : si les objets sont présents dans l'image ont des couleurs bien distincts et uniforme, ils vont apparaitre comme des pics dans l'histogramme, et chaque pixel est décrit selon certain canal : R, G, B, H, S, V,…ces algorithme travaillent sur plusieurs histogrammes, un par Channel comme l'algorithme RHS (Recursive Histogramme Splitting).

Ces approches sont très rapides et peu sensibles au bruit, mais ils ignorent les informations de proximité qui permettent d'utiliser des seuils variables locaux, ils ont une faiblesse dans les images qui ont la même couleur.

8 .1.2 Approches LOCALES : elles contiennent deux types de segmentation :

a. Segmentation basée sur les régions

Elle permet de grouper les pixels semblables, par exemples : seuillage /classification, méthodes d'agrégation, accroissement de régions …

Dans l'étape suivante on présente quelques algorithmes qui utilisent les graphes comme un outil de structuration dans cette approche :

a.1 Segmentation par germe

On part d'un point amorce et l'on étend en ajoutant les points de frontières qui satisfaits de l'homogénéité. Le point d'amorce peut être choisi par un humain, soit de manière automatique en évitant les zones de fonts contraste (gradient important) méthode d'amorce. Si le critère d'homogénéité est local, on fait la comparaison de la valeur du pixel de la frontière (méthode linière).

a.2 La ligne de partage des eaux (water shed)

Cette approche consiste à faire grossir simultanément toutes les régions jusqu'à ce que l'image soit entièrement segmentée. On considère les valeurs d'intensité des pixels d'une image comme une information d'altitude. Dans ce cas on peut représenter l'image (appelé carte d'évaluation) comme un terrain de trois dimensions, le principe est de remplir l'image progressivement d'eau. Chaque bassin du terrain représente une région.

L'algorithme se décompose en deux étapes :

- Générer une carte d'élévation à partir de l'image de départ.
- Remplir l'image progressivement les bassins.

b. Segmentation basée sur les contours

Elle permet de rechercher les pixels dissemblables (conteurs entre zones homogènes), espace échelle, modèles dérivatifs. Il y'a plusieurs algorithme utilisé dans cette approche mais on présente les fameux algorithmes :

b.1Les contours actifs (snake)

L'idée est d'utilisé des courbes déformables qui sont attirées par les formes recherchés dans l'image, et le « snake » se contracte et s'adapte à la forme.

b. 2 Algorithme de ConDensAtion :(Conditional Density propgation)

Est un algorithme de type (contours actifs), est un algorithme probabiliste intégrant :

- des informations contextuelles (modèle observationnel adapté au problème).
- Des connaissances à priori.
- Une intégration temporelle.

Ces approches sont très rapides à manipuler, et conceptuellement simple, mais ils n'ont aucune vision globale du problème. En pratique il y'a presque toujours un chemin continu de points connexes de couleur proche qui relier deux points d'une image (problème de gradient), et ils sont sensible au brui et peu stable.

8.1.3 Approches HYBRIDES

Est une segmentation accumule les deux approches précédents. On présente quelques algorithmes qui utilisent les graphes comme un outil de structuration :

a. L'algorithme de split and merge

Au plutôt que de regrouper les pixels de la région growing, pourquoi ne pas regrouper les zones homogènes pré-calculer sur l'image ? Cet algorithme décompose en deux parties :

a.1 Split

Créé les zones homogènes telles que l'image complète est la racine de l'arbre puis récursivement chaque feuille F est subdivisée en quatre si elle n'est pas assez homogène, et les quatre sous images sont ajoutés en tant que feuille de F. l'algorithme poursuit tant qu'il reste des feuilles non homogènes à diviser. On fait la construction du RAG (Region Adjancy Graphe) par la connectivité des régions adjacents, et les arrêtes sont mesures de différence d'homogénéité.

a.2 Merge

Chaque nœud du RAG (Region Adjancy Graphe) est examiné.si un des voisins de ce nœud à est une distance inferieur a un seuil de regroupement, les deux nœuds fusionnent dans le RAG.Lorsque plus aucun nœud ne peut fusionner avec l'un de ses voisins, stop.Cet algorithme permet de contrer le problème de gradient, mais il est assez complexe et le découpage un peu « carré », du à la topologie du quadtrees **[21].**

b. l'algorithme de CSC (Color Structur Code)

Est un algorithme Merge /Split basé sur topologie hiérarchique hexagonale avec recouvrement, l'idée est de regrouper des structures de pixels qui se chevauchent et découper ensuit les zones communes par une descente récursive.il décompose en trois phase :

b.1 Phase1 : Initialisation

Localement pour chaque ilot de niveau 0, on construit une partition en élément de niveau 0.

b.2 Phase2 : Regroupement

On se place dans un ilot de niveau n+1, on considère les éléments de niveau n contenu dans les ilots de niveau n, et les éléments de niveau n sont regroupés en de niveau n+1 si : ils sont

de couleur proche, ils se touchent. Les éléments sont structurés sous forme d'arbre, tout comme les ilots.

b.3 Phase3 : Découpage

Soient deux éléments E1 et E2 de niveau n, et S appartient à la fois à E1 et E2, le problème : S est attribuée à E1 ou E2 ? Attribution à celui dont la couleur est la plus proche. Si S n'est pas complètement attribué on fait un découpage récursif.

Cet algorithme permet de contrer le problème de gradient, et il a une excellente découpe des frontières des zones, mais il est très complexe.

8.2 .Recherche par contenu

Dans le contexte de la recherche par contenu d'images dans des bases de données, lorsque l'utilisateur formule une requête visuelle, sa cible de recherche est rarement représentée par une image entière comme le suppose le paradigme classique de recherche par une image exemple. L'image ne doit pas être traitée comme une unité atomique, car elle est généralement constituée d'un ensemble composite de zones visuelles exprimant une certaine sémantique. Dans ce contexte, les graphes, de par leurs natures, proposent une solution ajustée à ce problème. Bien évidement, l'utilisation des graphes augmente radicalement la complexité des tests d'évaluation de similarités entre les entités, mais les récents et rapides développements des techniques d'indexation multimédia ont proposé des solutions pour réduire la complexité d'appariement. Berretti et al ont proposé une technique d'appariement et d'indexation dédiée aux modèles de graphes dans la recherche par contenu. En fait, les images sont représentées par des graphes relationnels attribués "Attributed relational graph (ARG)" (image = < E, a, w>, où E respectivement présente l'ensemble des nœuds, a les attributs de nœuds et w les attributs des arcs appelés "relations spatiales binaires"). Ensuite, Berretti propose un modèle de comparaison des "ARG" avec trois distances (distance d'attribut, distance de relation et distance commune). En adaptant la technique d'indexation m-tree, d'autres solutions de recherche par composantes visuelles font recours aux graphes, notamment les graphes d'adjacence des régions "Regions Adjacency Graph (RAG)" également utilisé pour la reconnaissance de symboles dessinés à la main dans des documents de plans architecturaux. En l'occurrence, d'autres types de graphes ont été élaborés dans les travaux sur la recherche par le contenu comme le graphe de variance spatiale et le graphe d'adjacence des couleurs modifiés.

Certainement, la représentation sous forme de graphes est une solution adéquate pour la recherche par contenu d'images. Cependant, la problématique principale est le choix des informations à stocker dans ces graphes, par conséquence, les informations sont sélectionnées selon le type d'images à traiter. Par ailleurs, la complexité des algorithmes d'appariement présente un handicap pour l'utilisation plus générale des graphes **[22].**

8.3 Appariements de graphes et Similarités

On s'intéresse dans cette partie au problème de mesurer la similarité d'objets décrits par des graphes, et nous introduisons par une mesure de similarité de graphes. Cette mesure permet d'identifier et expliquer les différences et points communs entre deux graphes. Elle permet également de comparer des graphes décrivant des objets à différents niveaux de granularité car elle est basée sur des appariements multivoques (permettant d'associer un sommet d'un graphe à un ensemble de sommets de l'autre graphe). Enfin, cette mesure est générique et elle est paramétrée par deux fonctions qui permettent d'intégrer des connaissances spécifiques à l'application. Nous montrons que ces deux fonctions permettent de retrouver les différents types d'appariement de graphes, ainsi que d'autres mesures de similarité proposées plus récemment.

Nous nous intéressons ensuite par le calcul de cette mesure, et nous proposons et comparons quatre algorithmes différents :

– un algorithme glouton, qui construit de manière incrémental un appariement en choisissant à chaque itération le « meilleur » couple de sommets à ajouter à l'appariement en cours de construction.

Enfin, la section s'intéresse à la résolution d'un problème d'appariement de graphes particulier, à savoir le problème de l'isomorphisme de graphes. Ce problème, qui est bien résolu par des approches dédiées tirant partie d'invariants structurels pour élaguer très efficacement la recherche, devient en revanche très difficile à résoudre lorsqu'on le transforme en un problème de satisfaction de contraintes pour utiliser la programmation par contraintes car la transformation décompose la sémantique globale du problème initial en un ensemble de contraintes binaires. Afin de résoudre efficacement les problèmes d'isomorphisme de graphes avec la programmation par contraintes, nous introduisons une contrainte globale dédiée à ce problème et un algorithme de filtrage associé.

8.3 Similarité basée sur des appariements multivoques

8.3 .1.1 Caractéristiques communes à deux graphes relativement à un appariement

Afin de mesurer la similarité entre deux objets, il est courant et assez intuitif de comparer la quantité de caractéristiques communes à ces deux objets par rapport à l'ensemble de toutes leurs caractéristiques **[Tve77, Lin98]**. Un graphe G étant décrit par l'ensemble ***descr(G)*** des caractéristiques de ses sommets et arcs, la similarité de deux graphes G1 = (V1, E1) et G2 = (V2, E2) dépend des caractéristiques communes des ensembles ***descr(G1)*** et ***descr(G2).*** Cependant, étant donné que $\boldsymbol{V1 \cap V2 = \emptyset}$ l'intersection de ces deux ensembles de descripteurs sera toujours vide. Nous devons donc calculer cette intersection par rapport à un appariement m associant les sommets de G1 à ceux de G2.

Cet ensemble contient toutes les caractéristiques des éléments de G1 (resp. G2) retrouvées au moins une fois dans G2 ***(resp G1)*** via l'appariement m.

8.3.1.2 Similarité de deux graphes relativement à un appariement

Nous pouvons alors définir la similarité entre G1 et G2 en fonction de leurs caractéristiques communes par rapport à l'union de leurs caractéristiques :

$$\boldsymbol{sim1_m(G1, G2) = \frac{f(descr(G_1) \cap_m desc(G_2))}{f(descr(G_1) \cup descr(G_{2}))}}$$

Où f est une fonction positive croissante et monotone par rapport à l'inclusion, et telle que $\boldsymbol{f(\emptyset) = 0}$.Cette fonction permet de pondérer l'importance des différentes caractéristiques.

Notons que cette similarité est toujours comprise entre 0(quand $\boldsymbol{descr(G_1) \cap_m descr(G_2) = \emptyset}$ et 1 (quand$\boldsymbol{descr(G_1) \cap_m descr(G_2) = descr(G_1) \cup descr(G_2)}$) et qu'elle dépend de l'appariement m considéré : plus m apparie de composants (sommets ou arcs), et plus la similarité se rapproche de 1.

Cette première définition de la similarité n'est pas entièrement satisfaisante dans le sens où elle ne tient pas compte du fait qu'un sommet peut être « éclaté », i.e., apparié à plusieurs sommets. Il s'agit de prendre en compte ces éclatements de sommets dans notre mesure. Pour

cela, nous introduisons la fonction ***splits*** qui retourne l'ensemble des sommets « éclatés » avec l'ensemble des sommets auxquels ils sont reliés:

$\boldsymbol{split(m) = \{(v, s_v)|v \in V_1 \cup V_2, s_v = m(v), |m(v)| \geq 2}$ Et nous définissons un opérateur d'inclusion sur ces ensembles de sommets éclatés :

$$\boldsymbol{splits(m1) \subset splits(m2) \Leftrightarrow \forall (v, s_v) \in splits(m1), \exists (v, s'_v) \in splits(m_2), s_v \subset s'_v}$$

Nous pouvons alors modifier l'équation afin qu'elle prenne en compte les éclatements de sommets :

$$\boldsymbol{sim_m(G_1, G_2) = \frac{f\big(descr(G_1) \cap_m descr(G_2)\big) - g(splits(m))}{f(descr(G_1) \cup descr(G_2))}}$$

Où la fonction g possède les mêmes propriétés que f : elle est positive et croissante par rapport à la relation d'inclusion$\subset$ et $\boldsymbol{g(\emptyset) = 0}$.Dès lors, la similarité ne peut que décroître lorsque le nombre d'éclatements de sommets dans l'appariement m croît.

8.3.1.3 Similarité de deux graphes

Nous avons défini la similarité de deux graphes par rapport à un appariement entre leurs sommets. Maintenant, nous pouvons définir la *similarité* de deux graphes G1 et G2 comme la plus grande similarité par rapport à tous les appariements possibles: $sim(G_1, G_2) = \max_{m \subset V_1 * V_2} \frac{f(descr(f(descr(G_1) \cap_m descr(G_2)) - g(splits(m))G_1}{f(descr(G_1) \cup descr(G_2))}$

Notons que, si le degré de similarité $\boldsymbol{sim_m}$ obtenu pour un appariement m ne contenant pas de sommet éclaté est toujours compris entre 0 et 1, $\boldsymbol{sim_m}$ peut devenir négatif lorsque beaucoup de sommets sont éclatés : le poids de la fonction ***splits***, défini par g, peut alors être plus grand que le poids des caractéristiques communes, défini par f. Cependant, dans tous les cas, la similarité maximum sera toujours comprise entre 0 et 1 car la similarité obtenue par l'appariement vide est nulle.

Notons aussi que c'est le choix des fonctions f et g qui permet de déterminer le meilleur appariement. f et g seront donc choisis en fonction du domaine d'application et du type d'appariement recherché ; cet aspect est illustré dans les trois sections suivantes.

8.3.1.4 Connaissances de similarité

La mesure de similarité définie par l'équation est générique dans le sens où elle est paramétrée par les deux fonctions f et g : ces deux fonctions définissent l'importance relative des caractéristiques les unes par rapport aux autres. Le choix de f et g permet donc l'introduction de connaissances de similarité propres au domaine de l'application considérée. Ces fonctions sont souvent définies comme une somme pondérée. On introduit pour cela les trois fonctions de pondérations suivantes, définissant les poids des caractéristiques de sommets et d'arcs et les poids des éclatements :

$$\boldsymbol{poids_V} : \boldsymbol{V \times L_V \rightarrow R^+}$$

$$\boldsymbol{poids_E} : \boldsymbol{V \times L_E \rightarrow R^+}$$

$$\boldsymbol{poids_G} : \boldsymbol{V \times \partial(V) \rightarrow R^+}$$

On définit alors les fonctions f et g par :

$$\boldsymbol{f(F)} = \sum_{(u,l)\in F} \boldsymbol{poids_V}\,(v,l) + \sum_{(V_1,V_2,l)\in F} \boldsymbol{poids_E}\,(v_1,v_2,l)$$

$$\boldsymbol{g(S)} = \sum_{(v,s_v)\in S} \boldsymbol{poids_g}\,(v,s_v)$$

Sans connaissances particulières du domaine, les fonctions de pondération ***poids*** et ***poids*** peuvent retourner une valeur constante, ce qui revient à considérer que toutes les caractéristiques sont également importantes. La fonction poids g (v, s_v) peut retourner une valeur proportionnelle à la cardinalité de s_v.

A contrario, si des connaissances contextuelles sont disponibles, il est possible d'associer un poids particulier à chacune des caractéristiques et à chacun des éclatements de sommets.

8.4 Algorithmes pour mesurer la similarité de graphes

Etant donnés deux graphes multi-étiquetés G1 et G2, on considère maintenant le problème de calculer leur similarité maximum, i.e., trouver l'appariement m qui maximise l'équation. Notons que le dénominateur $\boldsymbol{f\big(descr(G_1) \cup descr(G_2)\big)}$ de cette équation ne dépend pas d'un appariement : il est introduit afin de normaliser le degré de similarité entre zéro et un. Il est donc suffisant, pour déterminer la similarité maximum entre deux graphes, de trouver l'appariement m qui maximise la fonction :

$$score(m) = f(descr(G_1) \cap_m descr(G_2)) - g(splits(m))$$

Ce problème est fortement combinatoire : il est plus général que le problème NP-difficile de recherche de plus grand sous-graphe commun.

Ce problème peut être formulé sous la forme d'un problème de satisfaction de contraintes valué: on associe une variable à chaque sommet de G1, son domaine étant l'ensemble des parties de l'ensemble des sommets de G2 , les contraintes expriment le fait que chaque caractéristique de G1 et de G2 doit appartenir à l'ensemble des caractéristiques communes aux deux graphes ***descr(G1)*** et ***descr(G2)*** , la valuation des contraintes est définie par les fonctions de pondération. Cependant, dans cette modélisation, le domaine de chaque variable comporte $\mathbf{2}^{V2}$ éléments de sorte que la recherche risque fort de ne pas terminer en un temps « raisonnable ».

L'espace de recherche du problème étant composé de tous les appariements possibles des deux graphes, c'est-à dire de tous les sous-ensembles de $V2 * V1$ .Malheureusement, la fonction score n'est pas monotone par rapport à l'inclusion, i.e., le score d'un appariement peut diminuer ou augmenter quand on lui ajoute un nouveau couple de sommets. Cela vient du fait que la fonction score est définie comme une différence de deux fonctions toutes deux croissantes lors de l'ajout d'un nouveau couple.

Ainsi, ce problème d'appariement de graphes apparait difficile à résoudre par une approche complète, et nous utilisons dans cette partie trois algorithmes : un algorithme glouton, un algorithme tabou, un algorithme réactif. L'algorithme de glouton a été implémenté mais les deux autres algorithmes ne sont pas implémentés car il n'y a pas beaucoup de temps.

8.4.1 Recherche gloutonne

Nous présentons un algorithme glouton proposé par Sébastien Sorlin et Christine Solnon pour apparier deux graphes donnés

a. Processus

L'algorithme glouton démarre d'une solution vide. Il ajoute itérativement à l'appariement m un couple de sommet tel que ce couple augmente le plus la fonction « score » ou bien la similarité de graphes. À chaque étape, les candidats à considérer sont des couples de sommets qui ne sont pas encore appariés (qui n'appartiennent pas à m) et qui améliorent l'appariement

actuel ***m***. Le couple à ajouter est choisi selon une heuristique gloutonne : l'ajout du couple doit maximiser la similarité. S'il y a plusieurs candidats maximisant la similarité, nous choisirons aléatoirement un parmi eux. L'algorithme s'arrête quand aucun couple de sommets ne peut augmenter la similarité en l'ajoutant à ***m***. À ce moment-là, l'ensemble de candidats est vide.

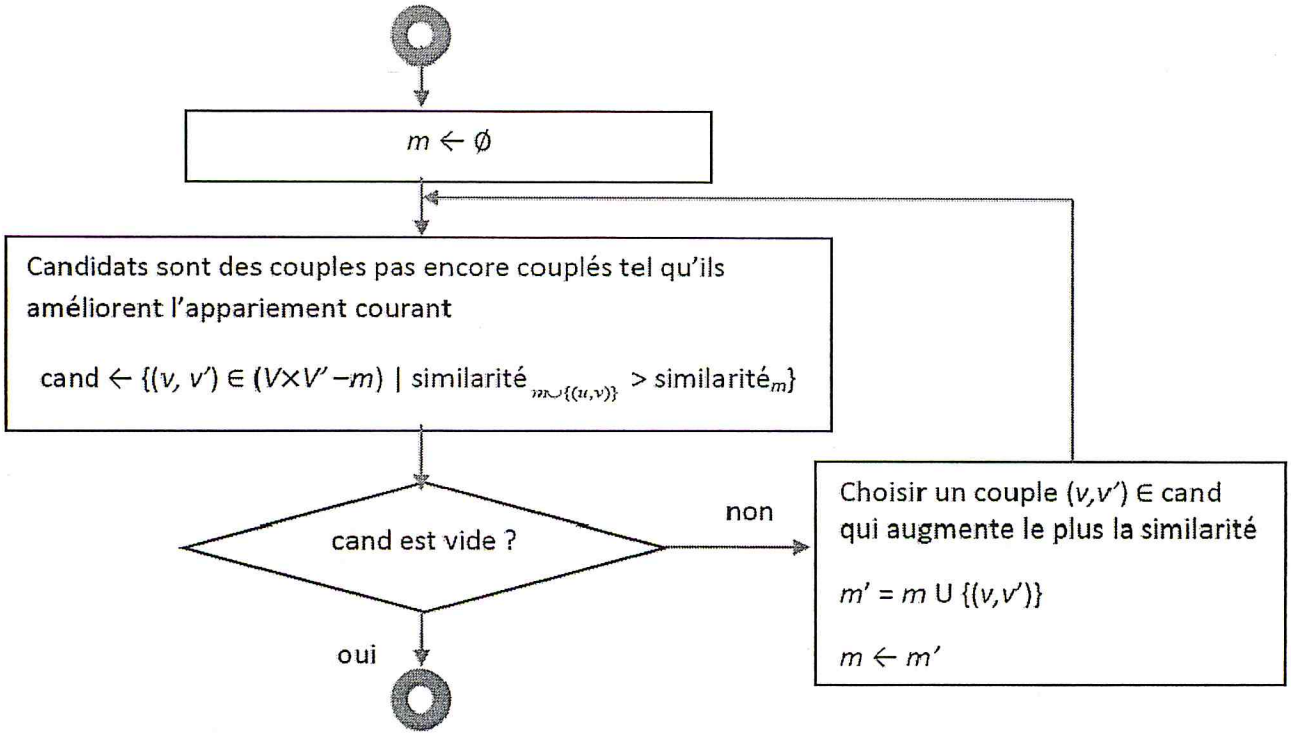

Figure II.4 : Diagramme de l'algorithme Glouton. [22]

Nous présentons dans la figure III.14 le diagramme de l'algorithme glouton pour illustrer le processus de la recherche gloutonne.

Le voisinage d'un appariement ***m*** se compose des appariements obtenus par l'ajout ou l'enlèvement d'un couple à ***m***. Les candidats considérés à chaque étape s'occupe donc une partie du voisinage. Par conséquent, l'algorithme glouton réduit encore l'espace de recherche par rapport aux autres algorithmes de recherche locale. Le temps d'exécution est donc très court. Cet algorithme est appelé glouton pour les raisons suivantes :

- Utilisation du sélecteur glouton qui choisi toujours la meilleure solution parmi les candidats
- Considération seule des solutions voisines améliorant l'appariement courant. Il n'accepte jamais une dégradation de qualité de la solution

Cet algorithme ne peut jamais retourner en arrière parce qu'il faut enlever certaines couples de sommets. De plus, il n'est pas complet bien qu'il puisse parfois trouver l'optimum global mais c'est aléatoire, pas systématique. Par conséquent, il est utilisé pour créer une solution initiale pour les autres algorithmes de recherche locale. Par ailleurs, cet algorithme n'est pas déterministe. Il obtient différentes solutions pour différentes exécutions. Nous pouvons l'exécuter plusieurs fois afin d'explorer des différentes zones de l'espace de recherche et garder la meilleure solution trouvée.

En utilisant le sélecteur glouton et un filtrage des meilleurs voisins, la qualité de la solution temporaire cherchée par l'algorithme glouton augmente toujours d'étape par étape, comme illustré dans la figure III.15.

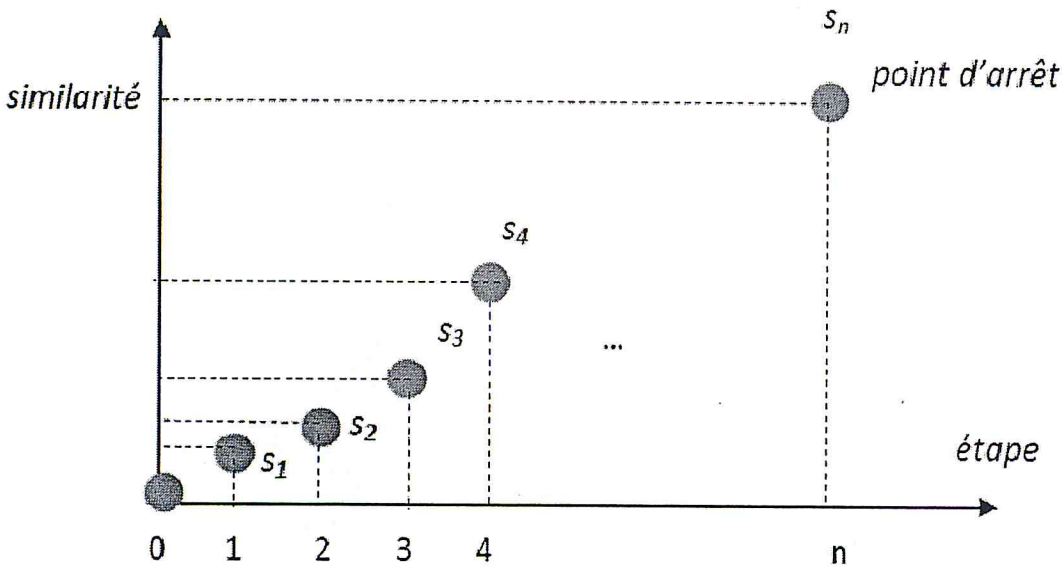

Figure II.5 : Evolution de la recherche par l'algorithme glouton. **[22]**

a. Algorithme

Fonction Algo-appariement$(G = (V, E), G2 = (V', E'))$ ***retourne*** *un appariement* $mC\, V \times V'$

$m \leftarrow \emptyset$

$cand \leftarrow \{(u, v) \in V \times V' | Similarité_{m \cup \{(u,v)\}}$ *Est le plus grand*

Tant que $cand ! = \emptyset$

Choisir $(u, v) \in cand$ *tel que* $Similarité_{m \cup \{(u,v)\}}$ *est le plus grand*

$m \leftarrow m \cup \{(u, v)\}$

$$cand \leftarrow \{(u,v) \in V \times V' | Similarité_{m \cup \{(u,v)\}} > Similarité_m\}$$

Fin tant que

Retourner ***m***

8.4.2 Recherche taboue

La recherche gloutonne retourne un appariement localement optimal dans le sens où il ne peut pas être amélioré en ajoutant ou en supprimant un seul couple de sommets.

Supposons une évolution de l'appariement ***m*** par la recherche gloutonne qui s'arrête après ***n*** étapes comme suit :

$$m_0 = \emptyset \rightarrow m1 \rightarrow m_2 \ldots \rightarrow m_n$$

Où ***mi*** est l'appariement obtenu dans l'étape i en ajoutant un couple à ***mi-1***. Car ***mi*** est meilleur que **mi-1** pour tout i :

$$m_n \overset{meilleur}{>>} m_{n-1} \overset{meilleur}{>>} m_{n-2} \overset{meilleur}{>>} \ldots \overset{meilleur}{>>} m_1 \overset{meilleur}{>>} m_0$$

Donc, l'enlèvement d'un couple à l'appariement final m_n diminue sa qualité. Par ailleurs, nous ne pouvons plus améliorer cet appariement car m_n est la solution finale obtenue par la recherche gloutonne.

Néanmoins, nous pouvons encore améliorer cet appariement en ajoutant ou en supprimant plusieurs couples de sommets. Une recherche locale essaie d'améliorer une solution en explorant son voisinage et en acceptant une dégradation des solutions. Par conséquence, nous pouvons utiliser l'algorithme glouton pour créer une solution initiale et l'améliorer par un autre algorithme de recherche locale.

Les solutions voisines d'un appariement sont les appariements obtenus en ajoutant ou en supprimant un couple de sommet à ***m***.

a. Processus

La recherche taboue démarre d'une solution initiale calculée par l'algorithme glouton. À chaque itération, elle s'intéresse aux solutions dans le voisinage Comme analysé ci-dessus, si ***m*** est un appariement localement optimal, tous ses voisins, même le meilleur, seront de moins bonne qualité de ***m***. Donc, avec le sélecteur glouton pour choisir le voisin à explorer, la recherche est cantonnée autour de l'optimum local. Par conséquent, pour échapper aux optimums locaux, nous devons éviter de faire une boucle sur un petit nombre de solutions en interdisant de retourner en arrière. Autrement dit, il faut défendre des couples de sommets récemment choisis à explorer en utilisant une liste taboue. Cette liste de longueur ***k*** mémorise

k derniers mouvements effectués correspondant aux k couples de sommets ajoutés ou supprimés. Tous les couples mémorisés dans cette liste sont interdits d'être ré-sélectionnés. Donc, cette liste permet à la recherche locale d'interdire un mouvement inverse à un mouvement récemment effectué (ajouter/supprimer un couple de sommet récemment supprimé/ajouté). Quand un couple de sommets est ajouté ou enlevé, il est interdit pendant k itérations, de l'itération actuelle i à l'itération $i + k$. Après l'itération $k + i$, ce couple est disponible à choisir. Par ailleurs, pour guider la recherche locale vers la zone contenant l'optimum global, l'algorithme tabou appliqué au problème de matching multivoque de graphes utilise une méta-heuristique « aspiration » : si un mouvement interdit permet d'atteindre un appariement de meilleure qualité que le meilleur appariement trouvée, il est quand-même accepté et réalisé. À chaque itération, la recherche taboue choisit un couple de sommets dans le voisinage à ajouter ou supprimer tel que : Soit il n'est pas interdit par la liste taboue, soit il peut créer un meilleur appariement par rapport à la meilleure solution trouvée. Et il est le meilleur candidat dans le voisinage.

b. Algorithme

Fonction Tabou $((G = (V, E), G' = (V', E'), K, \textbf{limiteQualité}, \textbf{maxMouv})$ ***Retourne*** *un appariement* $m \subset V \times V'$

{

$m \leftarrow \textbf{Glouton}(G, G')$; $best_m \leftarrow m$; $\textbf{nbMouv} = 0$

Tant que $\textbf{similarité}\ best_m(G, G') <$ *limiteQualité et* $\textbf{nbMouv} <$ *maxMouv*

Faire

$cand \leftarrow \{m' \in \textbf{voisinage}(m) | \textbf{similarité}_{m'} > \textbf{similarité}_{best_m}(G, G')\}$

Si $cand = \emptyset$ **alors** *// pas d'aspiration*

$cand \leftarrow \{m' \in \textbf{voisinage}(m) | \textbf{pas Tabou}(m, m', k)\}$

Fin si

$cand \leftarrow \{m' \in cand \,|\, m' \text{ est maximal}\}$

Choisir aléatoirement $m' \in cand$

$m \leftarrow m'; \textbf{Tabou}(m, m', k); \textbf{nbMouv} \leftarrow \textbf{nbMouv} + 1;$

Si

$\textbf{sim}_m(G, G') >$ *similarité*(G, G') $\textbf{alors}\ best_m \leftarrow m;$

Fin si

Fin tant que

}

Nous présentons dans la figure III.15 le diagramme de l'algorithme tabou pour illustrer le processus de la recherche gloutonne.

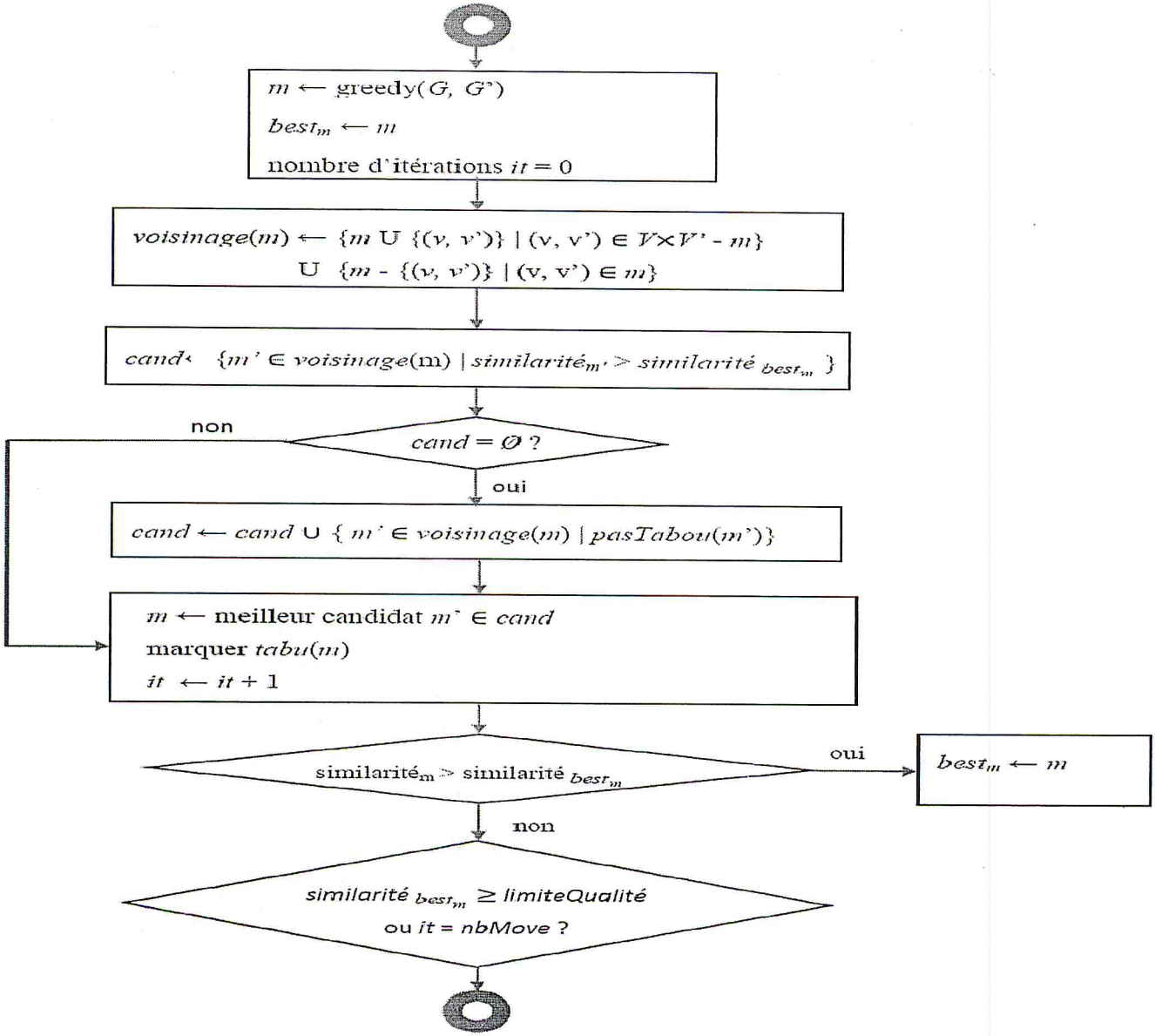

Figure II.6 : Diagramme de l'algorithme tabou. **[22]**

9. Conclusion

Nous pouvons conclure que les graphes sont une forme de représentation très polyvalente et flexible adaptée au domaine de l'analyse d'images. L'intérêt des graphes dans le traitement d'image. La flexibilité et la faculté à s'adapter aux différents type des données ainsi que la possibilité de traiter une sous partie sans avoir besoin de considérer une image comme un tout, toutes ces caractéristiques sont des atouts pour les approches basées sur les graphes dans le domaine de l'analyse d'images et la recherche d'informations notamment la recherche par le contenu d'images.

ChapitreIII : Conception d'un SRI basée sur les graphes

1. Introduction

Nous envisageons dans ce chapitre de proposer un système de recherche d'image basé sur les graphes. Ce système va utiliser une segmentation par germes qui est une des méthodes de graphes pour générer une signature sous forme de graphe d'une image donnée. Cet algorithme nous servira a indexer les images de la base et les images requêtes. Il sera question ensuite de recherche d'images. Pour une recherche efficace, il faut développer un moyen de trouver les graphes les plus similaires, ceux justement représentant les images similaires. Pour ca cette deuxième partie, nous avons utilisé un algorithme de d'appariement (l'algorithme glouton) basé sur le calcul de distance proposé par Sébastien Sorlin et Christine Solnon**[22].**

2. Algorithme de segmentation par germes

Cet algorithme permet de segmenter une image couleur, c'est-à-dire de diviser celle-ci en une mosaïque de régions. Il nécessite en entrée la matrice des pixels composant l'image et fournit en sortie une matrice indiquant pour chaque pixel le numéro du segment auquel il appartient.

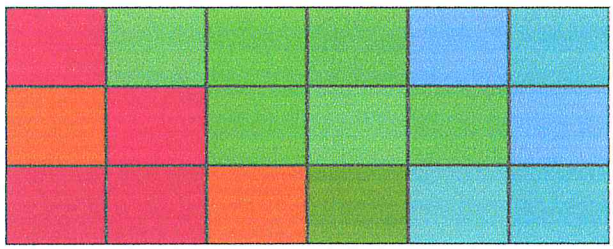

Figure III.1 ***Matrice d'entrée représentant l'image traitée***

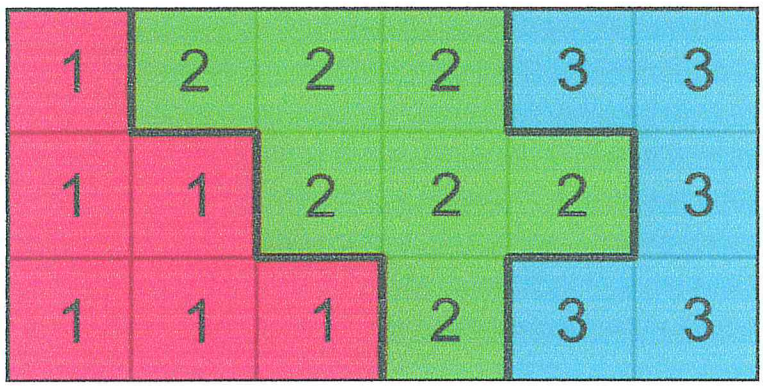

Figure III.2 ***Matrice de sortie (après segmentation) indiquant pour chaque pixel l'estampille du segment auquel il appartient***

La figure III.7 présente le résultat de la segmentation de l'image de la figure III.6. L'information décrivant la segmentation de l'image se résume donc à une matrice de dimension égale à celle de l'image.

2.1. Génération d'une région

1. Un pixel qui n'est associé à aucune région est choisi aléatoirement. Ce pixel forme le premier constituant de la région à venir et est appelé pixel germe ou simplement germe de la région. Cette dernière se sert de ce germe comme point d'ancrage.
2. La couleur du germe sert de couleur de référence pour la génération de la région.
3. Les quatre pixels voisins du germe (nord, sud, est, ouest) sont successivement testés. Ces derniers sont ajoutés à la région courante à la condition que leur couleur soit suffisamment proche de la couleur de référence et qu'ils ne fassent pas déjà partie d'une région. Sinon ils sont laissés de côté, pour être par la suite associés à une autre région. Deux couleurs sont considérées comme suffisamment proches l'une de l'autre si leur différence est inférieure à la valeur de seuil choisie (voir Choix du seuil de segmentation ci-dessous).
4. Les pixels voisins des pixels qui viennent d'être ajoutés sont à leur tour testés. Ceux qui passent le test sont ajoutés à la région. Cette itération se répète jusqu'à ce que tous les voisins aient été testés. Ainsi la région s'étend sur l'image comme une tache d'encre qui se répand sur une feuille, et dont l'expansion s'arête à la frontière formée par une autre tache antécédente.
5. A ce stade, la génération de la région est terminée. L'algorithme reprend alors son exécution au point 1 en vue de former une autre région. Lorsque tous les pixels de l'image appartiennent à une région, la segmentation est terminée.

Il est a noté que le choix de considéré les quatre voisins nord, sud, est et ouest n'est pas le seul possible. Un autre choix possible serait de tester les neuf voisins du pixel.

2.2. Choix du seuil de segmentation

Pour limiter la profondeur de l'expansion des régions, une valeur de seuil de segmentation doit être spécifiée avant de commencer l'algorithme. Cette valeur indique la différence maximum tolérée entre la couleur du germe et celle du pixel testé pour que ce dernier soit admis dans la région. Si la couleur des pixels qui composent l'image source est codée dans le format RGB, alors l'écart entre deux couleurs est calculé ainsi :

- **Formule :**

Couleur C1 = {C1r, C1v, C1b}

Couleur C2 = {C2r, C2v, C2b}

Ecart entre C1 et C2 = racine ((C1r-C2r)^2 + (C1v-C2v)^2 + (C1b-C2b)^2))

➢ **Exemple :**

Couleur C1 = {50, 50, 50} = gris foncé

Couleur C2 = {30, 50, 130} = bleu foncé

Ecart = racine ((50-30)^2 + (50-50)^2 + (50-130)^2) = 82

Dans le cadre de la segmentation par germes, un pixel est ajouté à une région si l'écart de couleur entre ce pixel et le germe de la région est suffisamment faible. Cette contrainte est respectée si cet écart est inférieur au seuil spécifié. Pratiquement, un seuil de 30 fournit pour la plupart des images traitées un bon compromis entre la finesse de la segmentation et la quantité de segments générés.

Pour des raisons pratiques, une autre valeur de seuil doit être spécifiée, à savoir celle du seuil de dé-segmentation.

2.3. Choix du seuil de dé-segmentation

Le résultat d'une segmentation contient généralement une grande quantité de régions dont la taille est négligeable. Un traitement approprié doit donc être effectué dans le but de supprimer ces régions. Le but du seuil de dé-segmentation est de définir une taille minimale sous laquelle les régions produites sont fusionnées à leur voisines. Ce traitement revient donc simplement à vérifier si la taille des régions générées est suffisante, et à fusionner celles dont la taille n'atteint pas le seuil imposé.

Pratiquement, un seuil de dé-segmentation égal au cinquième du seuil de segmentation offre dans la plupart des cas un résultat satisfaisant.

➢ **Exemple de segmentation**

Les figures III.3 à III.6 présentent pour une image codée en niveaux de gris (de -2 à +3) quatre segmentations pour quatre valeurs de seuil variant de 0 à 3. Le pixel germe de départ est le même pour les quatre images. Le niveau de seuil étant variable, la région s'étend plus ou moins, et reste même cantonnée à un unique pixel (celui du germe) dans le cas d'un seuil nul (figure III.6). Il est à rappeler que la couleur de référence utilisée est celle du germe, et pas celle des pixels voisins.

-2	-2	+2	+3
+2	0	-1	+3
+1	0	+1	-1
+2	-1	+2	+1

Figure III.3 image en niveau de gris. Seuil = 0

-2	-2	+2	+3
+2	0	-1	+3
+1	0	+1	-1
+2	-1	+2	+1

Figure III.4 image en niveau de gris Seuil = 1

-2	-2	+2	+3
+2	0	-1	+3
+1	0	+1	-1
+2	-1	+2	+1

Figure III.5 image en niveau de gris. Seuil = 2

-2	-2	+2	+3
+2	0	-1	+3
+1	0	+1	-1
+2	-1	+2	+1

Figure III.6 image en niveau de gris Seuil = 3

Finalement, les figures III.7 et III.8 présentent la segmentation obtenue pour un seuil de deux sur une image en niveaux de gris (de 0 à 8). Quatre germes ont suffit pour couvrir l'image de régions.

0	1	1	1	3	4	5	5	5	5
1	0	3	2	3	4	4	0	1	1
0	0	3	2	1	4	5	0	1	0
6	6	3	2	2	2	5	1	0	0
7	7	4	4	4	4	5	1	0	0
8	7	7	7	7	1	1	2	0	0

Figure III.7 ***Image source en niveaux de gris.***

0	1	1	1	3	4	5	5	5	5
1	0	3	2	3	4	4	0	1	1
0	0	3	2	1	4	5	0	1	0
6	6	3	2	2	2	5	1	0	0
7	7	4	4	4	4	5	1	**0**	0
8	7	7	7	7	1	1	2	0	0

Figure III.8 *Segmentation pour un seuil de 2 (germes en gras).*

➢ **Algorithme de Segmentation par germes :**

Définir une valeur positive pour le seuil.

Tant que (Il reste des pixels qui ne sont pas associés à une région) :

Générer un nouveau germe :

Trouver dans l'image un pixel auquel aucune région n'est associée (en parcourant l'image ou en choisissant des pixels de manière aléatoire).

Ce pixel devient le germe courant.

Générer la région associée au germe :

Attribuer une estampille à la région courante.

Marquer le germe avec l'estampille de la région courante; il fait désormais partie de la région.

Empiler le germe courant sur la pile.

Tant que (La pile n'est pas vide) :

Dépiler un élément de la pile.

Pour (Chacun des voisins de cet élément) :

Si (L'écart de couleur entre le germe et l'élément voisin est inférieur au seuil)

Et (L'élément n'est pas déjà marqué par l'estampille d'une région) :

Marquer l'élément voisin en lui attribuant l'estampille de la région courante.

Empiler l'élément voisin sur la pile; il sera ajouté à la région.

Sinon : Soit la couleur de l'élément voisin est trop différente de celle du germe, soit l'élément voisin fait déjà partie d'une région; il ne peut de ce fait pas être ajouté à la région courante.

A ce stade, la génération de la région courante est terminée.

3. Structure de donnée utilisée

3.1 Matrice de cases

Elle indique pour chaque pixel sa couleur, sa position, l'estampille du segment auquel il appartient, le pixel suivant du segment et le pixel précédent. Leurs connaissances permettent de former une liste doublement chaînée qui parcourt tous les pixels du segment ou région.

3.2 Tableau de segments

Il contient les caractéristiques de chaque segment, à savoir son estampille, sa case de départ, et ses caractéristiques secondaires; dans notre cas sa couleur moyenne et son aire.

- **Exemple**

Dans l'exemple qui suit, l'image source (figure 4) est une simple image de 3 pixels de large sur 2 de haut codée au format RGB (Red, Green, Blue). Une fois la segmentation terminée, la matrice de cases et le tableau de segments contiennent toute l'information concernant la segmentation de l'image source, à savoir :

1. le numéro de la région associée à chaque pixel.
2. les caractéristiques intrinsèques de chaque région (estampille, case de départ, couleur moyenne et aire);
3. la liste doublement chaînée permettant de relier entre eux tous les pixels d'une région.

Toutes ces informations sont représentées sur les figures 5 et 6.

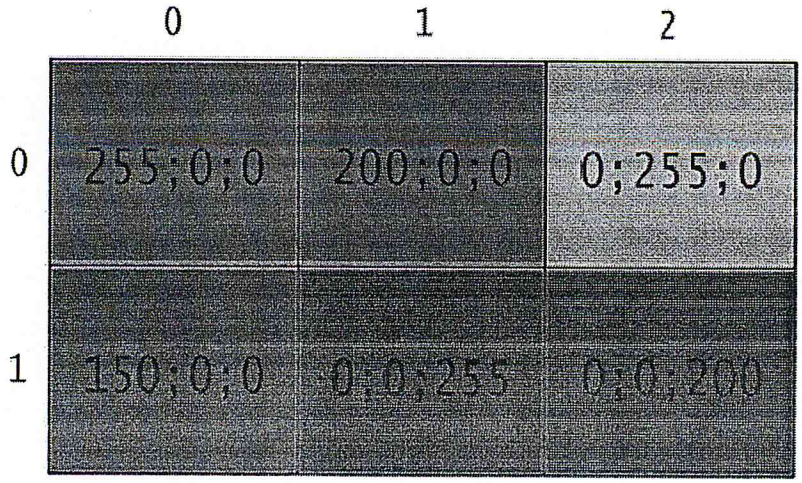

Figure. III.9 ***Image source***

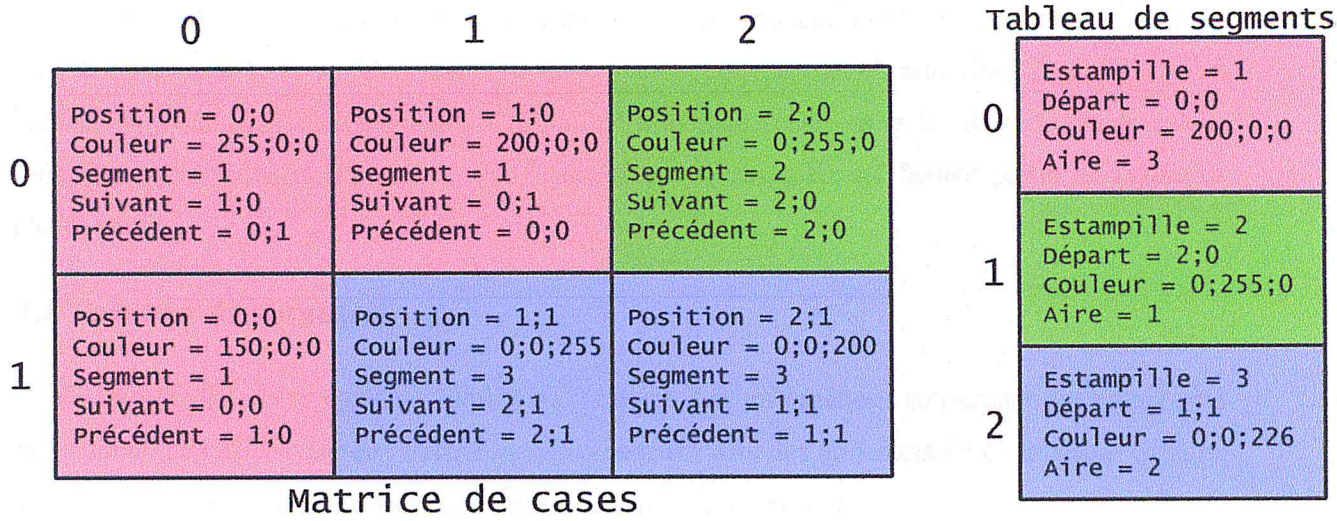

Figure III.10 *Matrice de cases et tableau de segments correspondant à l'image source*

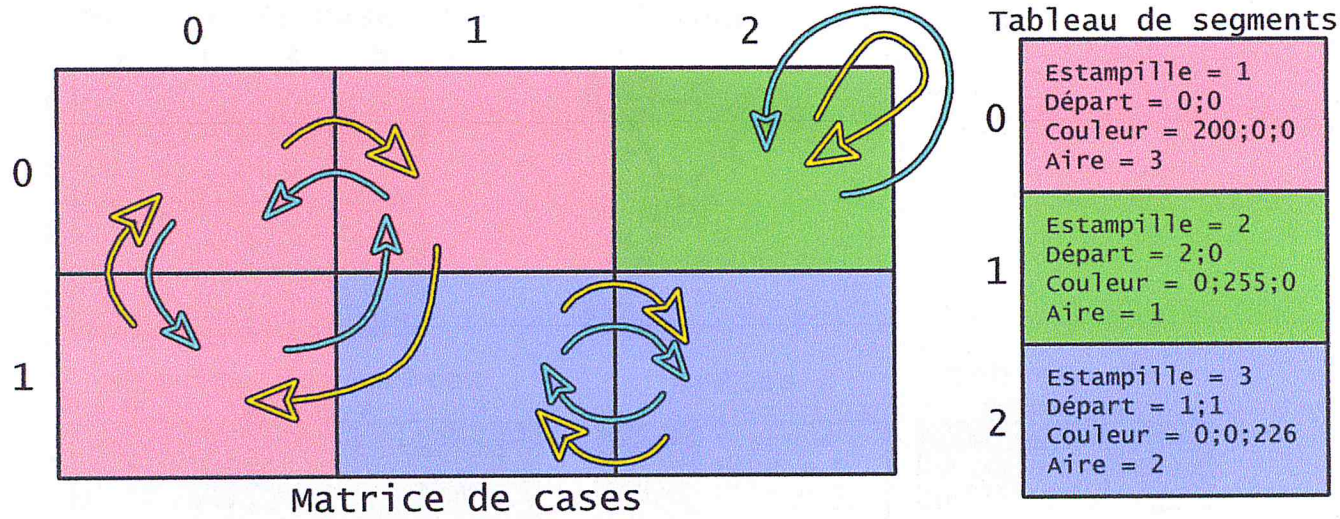

Figure III.11 ***Flèches jaunes : Liste chaînée des cases suivantes***

Flèches bleues : Liste chaînée des cases précédentes.

➢ **Explication**

Afin de construire les segments de l'image requête nous avons utilisé un algorithme qui visite les pixels qui ont le même numéro de région puis il assemble ces pixels ensuit on calcule la couleur moyenne (comme la figure de tableau de segment), si les pixels d'une telle région se terminent on calcule la couleur moyenne de la région et le nombre des pixels de cette région, puis on passe la région suivante jusqu'à la fin de toutes les régions.

3.3 Matrice des arêtes

Elle permet de savoir s'il existe une arête entre deux sommets donnés. Cette matrice est une matrice carrée de dimension égale au nombre de sommets qui composent le graphe. Ce nombre est lui-même égal au nombre de segments composant l'image segmentée. De plus, cette matrice est triangulaire du

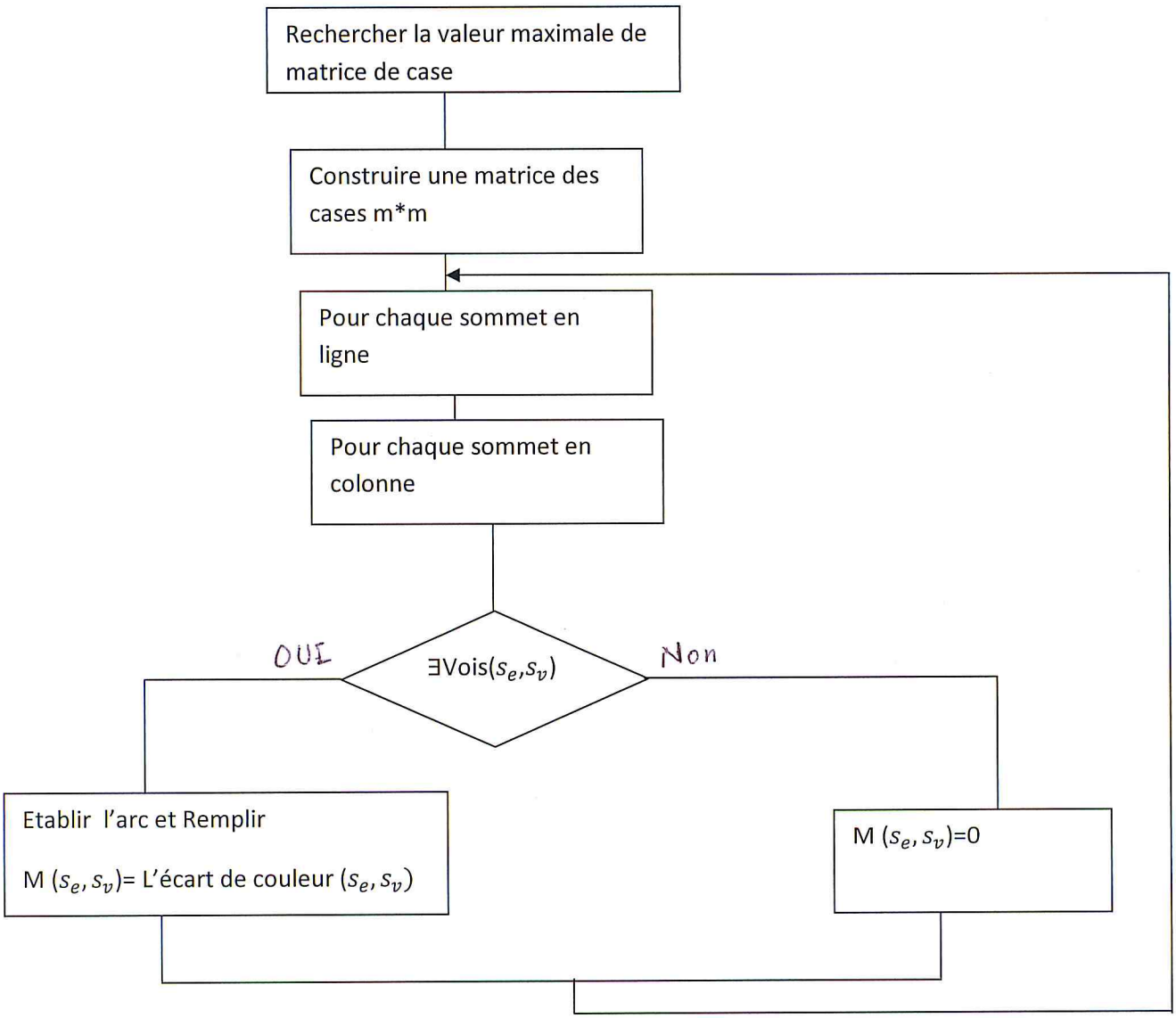

Figure III.1 3 : *diagramme représente le processus d'extraction d'un graphe.*

5. La conception de système

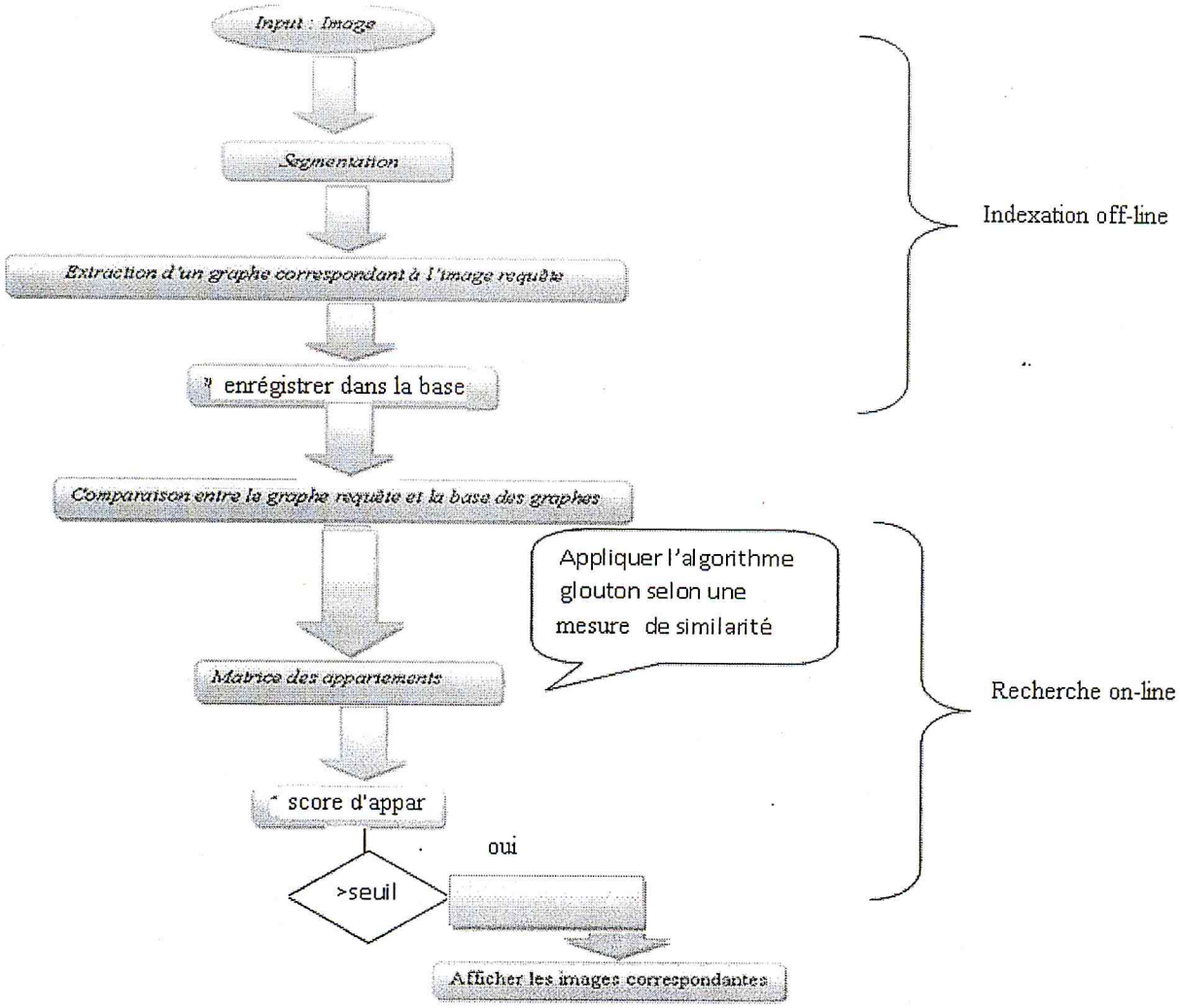

Figure III.13 : Le diagramme de processus proposé.

Conclusion

Nous avons vu dans ce chapitre les algorithmes utilisé dans ce projet, l'algorithme de segmentation par germe pour l'extraction du graphe correspondant à l'image requête et l'algorithme glouton pour résoudre le problème d'appariement de graphes ou bien de trouver les images pertinentes.

CHAPITRE 4 :
EXPIREMENTATION ET EVALUATION

1. Introduction

Après avoir effectué la conception de notre projet, nous allons à présent entamer la réalisation de ce dernier. Nous présenterons alors, dans un premier lieu, l'environnement de développement (langages et outils) ensuite, une évaluation des résultats et comparons la performance de cet algorithmes de recherche sur la qualité des solutions et sur le temps d'exécution nous présenterons quelques captures d'écran.

2. Outils informatique utilisé

Pour développer les algorithmes utilisés dans notre approche d'optimisation, nous avons utilisé le langage de programmation orienté objet JAVA. Ce langage est conçu avec l'approche orientée objet, de sorte qu'en Java, tout est objet à l'exception des types primitifs (nombres entiers, nombres à virgule flottante, etc.).

Notre choix est basé sur les caractéristiques de ce langage comme, entre autres, la portabilité et l'indépendance vis-à-vis des plateformes. De plus, Java offre une bibliothèque très riche de classes comme celles pour l'utilisation du réseau ou de l'interface utilisateur graphique. Le langage Java offre aussi le support pour l'utilisation des processus légers (*threads*).

2.1. Eclipse IDE

Est un environnement de développement intégré libre, le terme *Eclipse* désigne également le projet correspondant, lancé par IBM.

Il est extensible, universel et polyvalent, permettant potentiellement de créer des projets de développement mettant en œuvre n'importe quel langage de programmation.

Eclipse IDE est principalement écrit en Java à l'aide de la bibliothèque graphique SWT, d'IBM.

2.2. Les APIs

* JAI

Java Advanced Imaging, ou plus simplement nommé JAI, est ainsi compatible avec le cœur de Java en reprenant les mêmes interfaces et classes dès que possible. Ainsi, JAI redéfinit ses propres classes représentant une image, mais continue d'utiliser les classes

ColorModel, SampleModel et l'interface RenderedImage. Cela permet de pouvoir indépendemment remplacer les images Java2D par celles définit par JAI.

JAI apporte les avantages suivants par rapport à Java2D :

- De meilleures performances par l'utilisation du système pour les traitements (par appel de code compilé pour la plateforme à l'aide de JNI).
- La séparation des images en plusieurs zones (Tiles) en utilisant la classe TiledImage qui est le pendant JAI de la classe BufferedImage.
- La prise en charge de régions d'intérets (ROI) afin de n'effectuer des traitements que sur une région précise de l'image.

2. 3. La base d'image

Nous avons utilisé une base d'images qui enregistré dans un fichier.

3. Description de l'application :

Le Menu « sélectionner la base » permet de sélectionner la base d'images qui les images que vous voulez chercher.

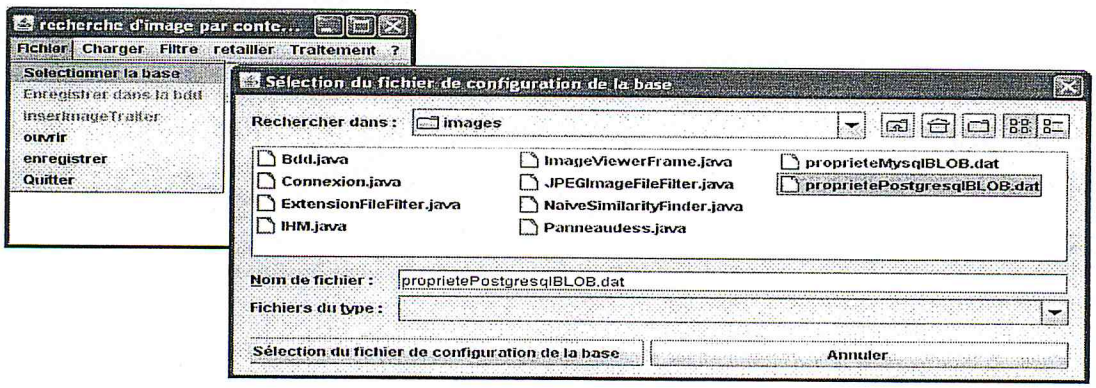

Figure IV.1 : Sélection de fichier de configuration de la base d'image.

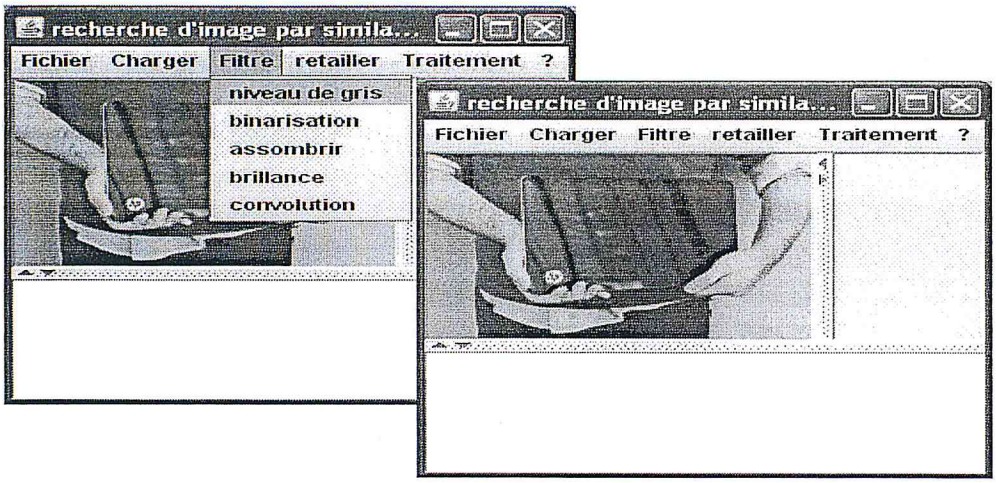

Figure IV.2 : Transformation d'une image en rgb à une image en niveau de gris.

Le menu « niveau de gris » permet de transformer une image en rgb à une image en niveau de gris.

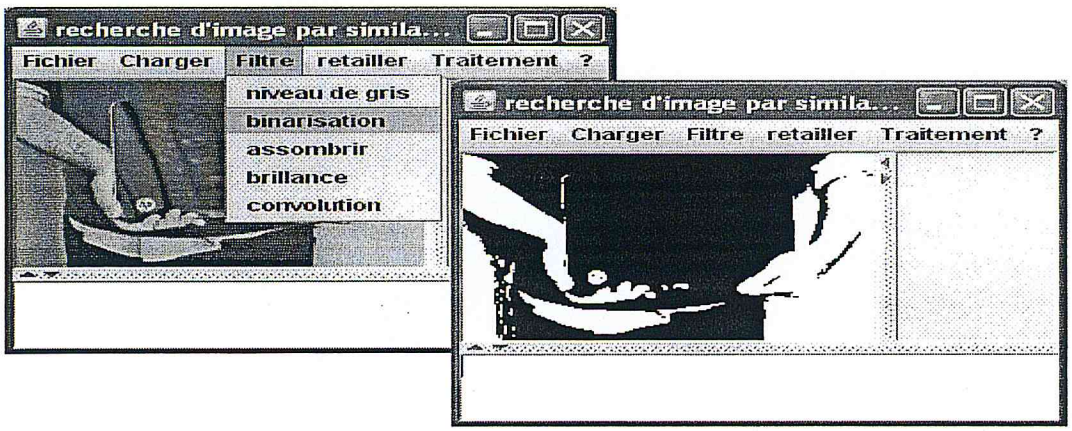

Figure IV.3 : Transformation d'une image en rgb à une image en binarisations.

Le menu « binarisations » permet de transformer une image en rgb à une image en binarisations.

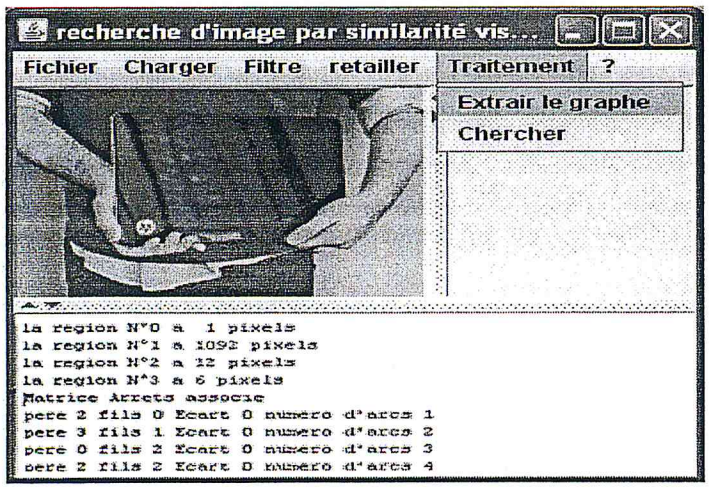

Figure IV.4 : Extraction d'un graphe correspondant à l'image requête.

Le menu « extraire le graphe » permet d'extraire un graphe correspondant à l'image requête.

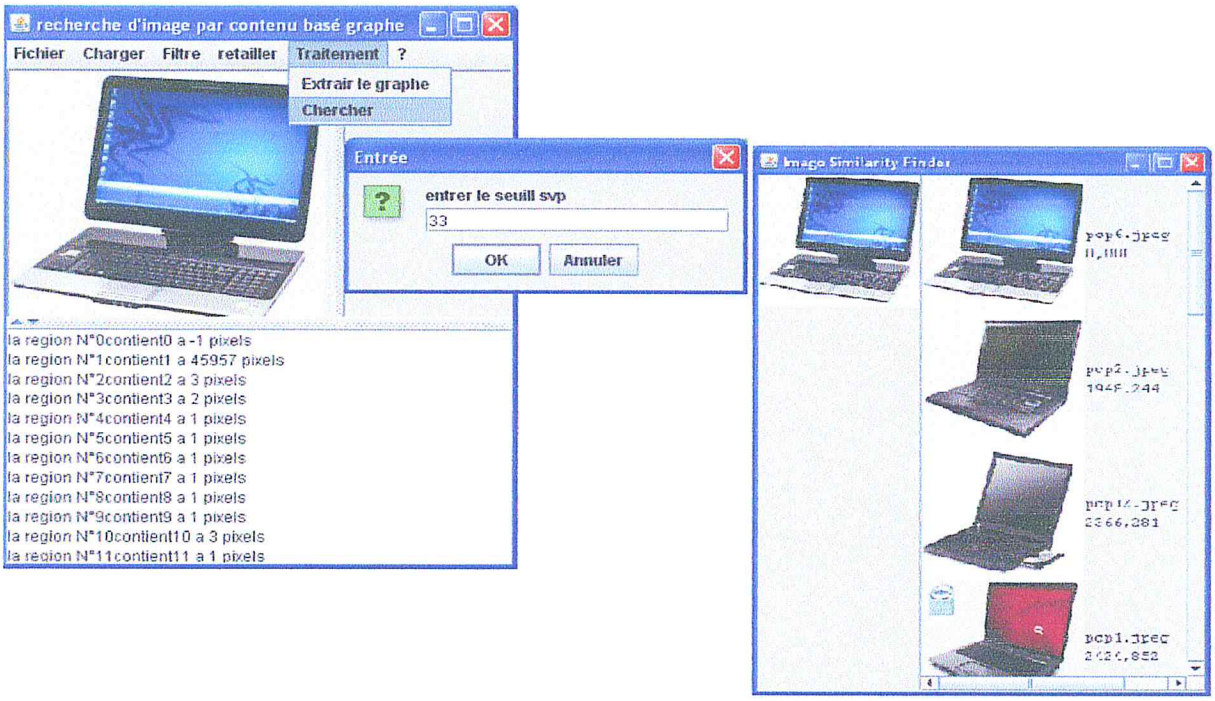

Figure IV.5 : la recherche des images correspondantes à l'image requête.

Le menu « Chercher » permet de chercher des images correspondantes à l'image requête qui fait une comparaison entre les graphes de base et le graphe requête.les images résultats sont similaire parce qu'elle ayant des graphes similaires. Donc la comparaison se fait entre les graphes correspondants de ces images par l'appariement multivoque et selon une mesure de similarité. Nous expliquerons mieux dans l'exemple suivant :

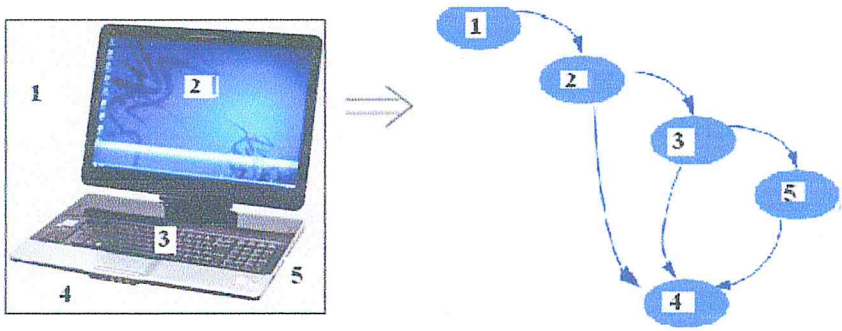

Figure IV.6 : Extraction d'un graphe à partir d'une image.

Les sommets et des arcs représentent respectivement des composants et les relations de position relative entre des composants de l'image.

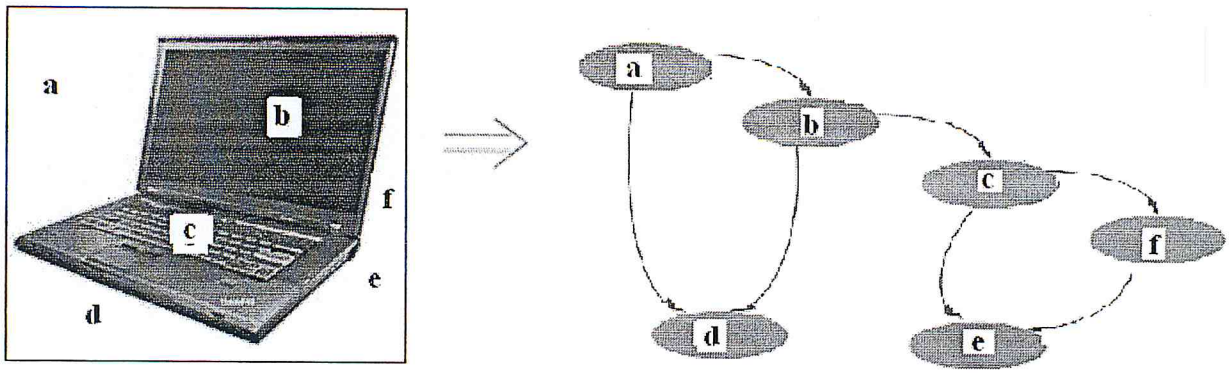

Figure IV.7 : Extraction d'un graphe à partir d'une image.

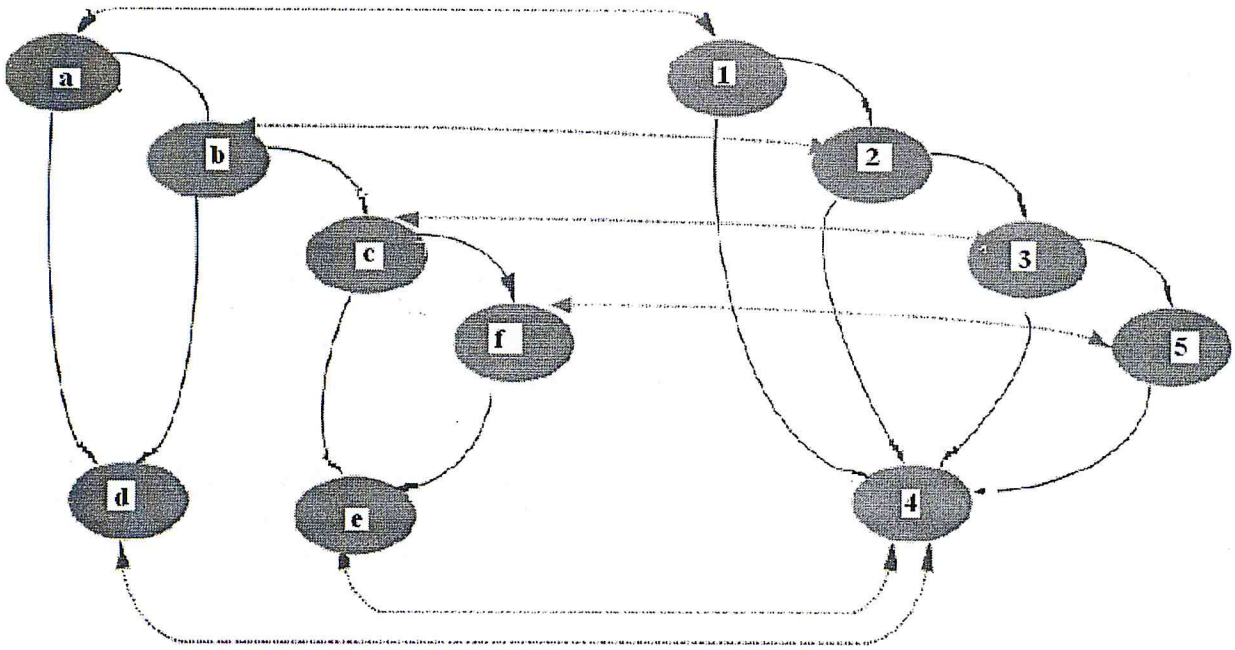

Figure IV.8 : Un appariement entre deux graphes.

Avec les deux graphes représentés dans la figure IV.4, il y a plusieurs appariements possibles. Nous définissons ci-dessous un matching de graphes: m = {(*a, 1*), (*b, 2*), (*c, 3*), (*d, 4*), (*e, 5*), (*f, 5*)} Cet appariement accouple respectivement les sommets a, b, c, d avec les *sommets 1, 2, 3, 4* et les sommets *e, f* sont tous les deux appariés au sommet *5*. Dans cet appariement, l'ensemble de sommets associés au sommet 5 est $m(5)$ = {*e, f*}. Si nous enlevons le couple (*f, 5*) de l'appariement m, nous obtenons un nouvel appariement : m' = {(*a, 1*), (*b, 2*), (*c, 3*), (*d, 4*), (*e, 5*)}.

Dans ce nouvel appariement, le sommet f n'est donc accouplé à aucun sommet du deuxième graphe. L'ensemble de sommets associés à f est vide : $m(f) = \emptyset$.

4. Test de validation :

Nous avons testé notre application sur un fichier d'images obtenons des résultats acceptables en général.

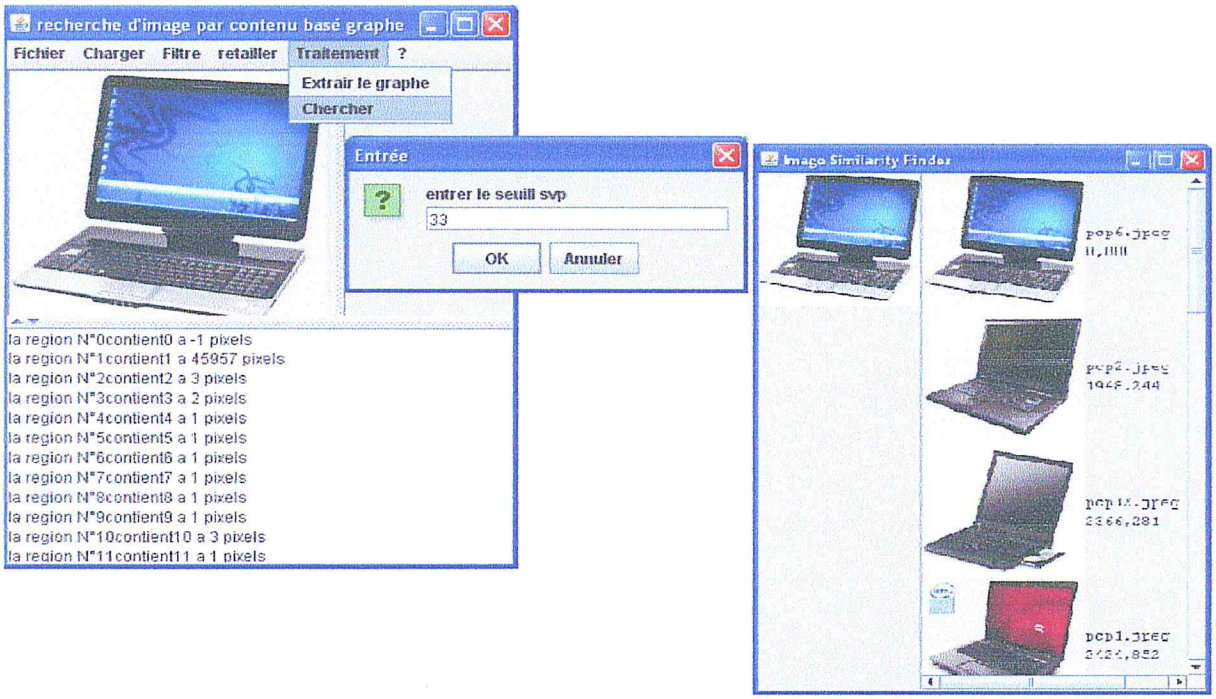

Figure IV.9 : la recherche des images correspondantes à l'image requête.

5. Conclusion

Les résultats obtenus nous sembles intéressants du bien que les images obtenus sont visuellement similaires. Ce que nous donne quelque garantie sur la qualité du système prototype.

Conclusion générale

Dans ce mémoire nous avons implémenté un moteur de recherche d'images par similarité visuelle basées sur les graphes et nous avons essayé d'améliorer le temps de recherche et la qualité des images pertinentes par l'utilisation des graphes et par l'application des algorithmes de recherches et d'appariement entre les graphes comme l'algorithme glouton afin de trouver des résultats intéressants et de trouver des images plus pertinentes. L'utilisation des graphes est une forme altérative au reçu d'images par similarité visuelle telle que l'a montré notre implémentation prototype. L'inconvénient majeur réside dans le lentement des algorithmes d'appariement.

Perspectives

Des études doivent être menues pour alléger les couts d'utilisation des graphes, c'est-à-dire en diminuer la complexité .Dans la mesure ou cela serait faisable, les graphes poursuivent être utilisés pour prendre en compte texture et forme également.

[1] : V.lozano Contribution de l'analyse d'image couleur au traitement des images textile. Thèse de doctorat, Université Jean-Monnet, Saint-Etienne, janvier 1998.

[2] : T.Gevers Color in mega databases.Technical report, University of Amsterdam, 2000.

[3] : CIE. Colorimétrie. Technical Report 15.2, Bureau central de la CIE, Vienna, 1986.

[4] : A.Manzanera .Cour de Ti-ENSTA .Unité d'électronique et informatique.

[5]: D.L MacAdam. Color measurement, theme and variation. Optical Science, Springer-Verlag 1985.

[6] : CIE. Colorimétrie. Technical Report 15.2, Bureau central de la CIE, Vienna, 1986.

[7] : VU Ngoc Son.Travail d'Intérêt Personnel Encadré (TIPE).

[8]: John P Eakins and Margaret E Graham. Content-based Image Retrieval. A report to the JISC Technology Applications Programme. Institute for Image Data Research, University of Northumbria at Newcastle. January 1999.

[9]: A. Smeulders, M. Worring, S. Santini, and A. Gupta. Content based image retrieval at the end of the early years. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(12):1349-1380, 2000.

[10]: T. P. Minka An Image Database Browser that Learns from User Interaction, Master of Engineering Thesis, 1996.

[11]: S. Staab, A. Maedche, Axioms are objects too: Ontology engineering beyond the modeling of concepts and relations, Research report 399, Institute AIFB, Karlsruhe, (2000).

[12] M. Flickher, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B.

Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D.Steele, and P. Yanker, "Query by Image and Video Content: The QBIC System," IEEE Computer, vol. 28, no. 9, pp. 23-32, Sept. 1995.

[13]: J. R. Smith and S.-F. Chang, Querying by color regions using the VisualSEEk content-based visual query system, In Intelligent Multimedia InformationRetrieval. IJCAI, pages 159-173, 1996.

[14]: T. S. Lai, "CHROMA: a photographic image retrieval system", PhD thesis, School of computing, engineering and technology, University of Sunderland, UK, 2000.

[15]: Remco C. Veltkamp, Mirela Tanase, "Content-Based Image Retrieval Systems: A Survey", Department of Computing Science, Utrecht University, Technical Report UU-CS-2000-34, March 8, 2001.

[16]: Guérin-Dugué A., Ayache S., Berrut C., Image retrieval : a first step for a human centered approach, in Fourth Pacific-Rim Conference on Multimedia (ICICS-PCM 2003), Singapore, 15-18 December 2003, 2003.

[17]: Applications des graphes en traitement d'images. Salim Jouili, Salvatore Tabbone ,LORIA - UMR 7503Université de Nancy 2Campus scientifique - BP 239 54 506, Vandœuvre -les-Nancy Cedex, France.

[18]: R. A. Wagner and M. J. Fisher, "The string-to-string correction problem". Journal of the ACM, Vol. 21, No. 1, pp. 168-173, 1974.

[19]: A. Robles-Kelly and E. Hancock, "Graph edit distance from spectral seriation", IEEE Transactions on pattern analysis and machine intelligence, Vol. 27, No. 3, pp. 365-378, 2005.

[21]: H. Bunke and K. Shearer, "A graph distance metric based on the maximal common subgraph", Journal of pattern recognition letters, Vol. 19. No. 3/4, pp. 255-259, 1998.

[20]: A. N. Papadopoulos and Y. Manolopoulos, "Structure-based similarity search with graph histograms", Workshop on database and expert systems applications, pp. 174-178,

1999.

[22]: Christine Solnon, "Contributions à la résolution pratique de problèmes combinatoires des fourmis et des graphes",l'Université Claude Bernard Lyon 1.