**Université de Nice Sophia-Antipolis**

**DEA SIC: Image-Vision**

**Rapport de Stage**

29 juin 2004

par Igor Rosenberg

# Utilisation de fonctionnelles de type rapport de coûts pour l'extraction de régions dans des images de télédétection

Lieu : Ariana, projet commun I3S/INRIA
Responsables du stage : Ian JERMYN et Josiane ZERUBIA

Année 2003-2004



# Table des matières

# Chapitre 1

# Introduction

## 1.1 Position du problème - but

Dans le cadre du sujet de stage de DEA, effectué dans l'équipe Ariana[1], on se propose d'extraire les regions correspondant à des zones urbaines dans une image satellite. On souhaite retourner des courbes fermées dont l'intérieur correspond à une partie de ville. Ce thème de recherche correspond à la demande d'analyse de zones urbaines, pour les besoins par exemple des télécommunications, ou la création de cartes d'occupation des sols, qui est à mettre en parallèle avec les modifications des villes ( par exmple, Macapá voit sa population augmenter de 10% par an).

La difficulté consiste à retrouver le contour exact, ainsi qu'à savoir exclure des zones particulières à l'intérieur des villes (parcs, lacs... ), dans un processus entièrement automatisé. De plus, la vérité de terrain n'est accessible qu'au travers d'un expert, ce qui complique le travail de vérification des résultats.

Le travail de stage repose sur deux méthodes développées au sein de l'équipe Ariana[1]. D'une part, les travaux de la thèse [11] d'Anne Lorette décrivent une méthode d'analyse statistique de la texture attribuant une valeur réelle à chaque pixel, valeur correspondant à un indice de confiance pour une zone urbaine. On utilise alors la méthode décrite par Jermyn et Ishikawa[7] (méthode de rapport de coût sur un bi-graphe), qui est capable d'extraire d'un graphe la région correspondant au minimum global d'une description à priori donnée sous forme d'énergie, pour définir où se trouvent exactement les frontières.

Le travail original apporté par le stage est de trouver des fonctions qui permettent de souligner les caractéristiques des villes, sur des images satellites, afin que la région trouvée par la méthode du rapport de coût corresponde à une zone urbaine.

## 1.2 L'approche rapport d'énergies

Dans le cadre de l'extraction de contours, on s'intéresse à minimiser une énergie $E = \frac{E_1}{E_2}$, rapport de deux fonctions d'énergie $E_1$ et $E_2$. Plus spécifiquement, on souhaite analyser la forme

$$E[R] = \frac{\int_R A(x)\mathrm{d}x}{\int_{\partial R} B(\gamma(s))\mathrm{d}s} \tag{1.1}$$

B est construite pour être toujours positive, et on souhaite que le dénominateur soit invariant par rapport à l'orientation de la courbe, tandis que A peut être négative, et doit changer de signe quand l'orientation change. Comme la valeur de cette énergie peut être négative, il est plus simple de voir le problème comme une une maximisation de la valeur absolue. On a ici l'inclusion de deux types de descripteurs de départ, en aboutissant à un équilibre entre la maximisation de la valeur absolue du numérateur et la minimisation du dénominateur (qui est toujours positif). On remarque que l'on n'a pas de paramètre qui relie les deux termes d'énergie. En effet la balance entre les deux termes est faite par le quotient, et la minimisation de l'un et la maximisation de l'autre.

Pour donner un exemple, $E_1$ peut être choisie comme une fonction du gradient de la région, tandis que $E_2$ décrit le périmètre de la région, en choisissant $B = 1$. Il faut comparer cela à une écriture plus standard de la forme $E = E_1 + \alpha E_2$, où l'évaluation du paramètre prend un rôle prépondérant, car c'est lui qui indique quel est l'apport de chacune des énergies en jeu. Par exemple, lorque $\alpha$ a une grande valeur, la contribution de l'énergie 1 est minime, tandis que pour une valeur de $\alpha$ proche de zéro, la courbe subira plus l'influence de cette énergie.

En prenant des rapports d'énergies, on se retrouve avec une énergie qui est invariante à la multiplication. En effet, multiplier les données par une constante ne change rien, grâce au quotient

[1] INRIA Sophia-Antipolis `http://www-sop.inria.fr/ariana/`

d'intégrales. Cette forme d'énergie est donc adaptée aux changements d'illuminations que l'on modélise généralement par la multiplication par un facteur. L'intérêt est aussi que l'on n'est pas biaisé vers des régions plus ou moins grandes. En effet, l'énergie du numérateur fait grandir la frontière, tandis que celle du dénominateur fait rapetisser la frontière. La balance se fait en fonction des données sous-jacentes. Par exemple, l'énergie d'une région donnée serait la même qu'une version réduite de moitié de la même région, en supposant que l'on ait dans les deux régions les mêmes propriétés.

L'avantage principal de cette formulation réside dans l'existence d'un algorithme pour trouver la solution globale au problème de minimisation, dont le coeur est décrit dans 3.1.3.

### 1.2.1 Discrétisation

Le but est d'extraire d'une image une région qui minimise la forme d'énergie donnée en 1.1. Il existe plusieurs façon de résoudre le problème sur les images, qui est un domaine discret, alors que l'énergie est décrite de façon continue. On pourrait essayer d'utiliser des méthodes variationnelles habituelles, genre "level sets". Ce sera la seconde partie du stage (on pourra regarder la section 5). Pour la première partie, je me suis concentré sur l'utilisation d'une méthode qui permet de trouver le minimum global de la fonction d'énergie. En effet, en projettant l'image sur un graphe, on est alors capable de trouver le cycle qui minimise globalement l'énergie (1.1).

Les travaux de Jermyn et Ishikawa [7] se concentrent sur l'application d'un algorithme d'extraction du cycle de rapport minimal sur un graphe dont les arêtes possèdent deux poids (bi-graphe), lorsque les données initiales sont données par une image. On cherche le cycle $C$ dont le poids $w(C) = \frac{\sum(\lambda)}{\sum(\tau)}$ est minimal, où $\lambda$ et $\tau$ sont les poids du graphe.

L'algorithme présenté rend le minimum **global** sur tous les cycles du graphe. On est donc assuré d'avoir la meilleure solution de la modélisation que l'on a proposée. Il faut mettre cette propriété en parallèle avec les techniques habituelles variationnelles, où on trouve généralement des minima locaux, car on utilise une descente de gradient, ce qui lie très fortement la réponse à l'initilaisation. Pour la méthode utilisée ici sur les graphes, on n'a pas ce problème : il n'y a pas d'initialisation de la courbe, et on trouve toujours la région minimisant le critère.

Par contre, on doit trouver la fonction de projection des données, qui envoie une image sur un graphe. C'est là que la modélisation a lieu. Il faut trouver une fonction qui souligne les éléments importants de l'image, et qui atténue les bruits et les zones qui ne nous intéressent pas. En effet, l'algorithme sur les graphes rend bien le minimum global, mais il ne présente aucune modélisation du phénomène auquel on s'intéresse. Il faut donc construire son entrée pour que le minimum corresponde à ce que l'on souhaite extraire. Un grand soin doit être apporté à l'écriture des fonctions $A$ et $B$. On peut voir cette projection comme le fait que l'algorithme est indépendant des données initiales, et donc applicable à toute sorte de données initiales. En effet, si l'on est capable de trouver une fonction qui projette les données sur un graphe, on peut extraire de ce graphe la région minimisant le rapport d'énergie. L'algorithme en lui-même ne se sert que du graphe généré. On peut donc réutiliser cette méthode sur des images médicales ou des images d'une séquence vidéo, sous réserve que l'on parvienne à projetter les données initiales sur un graphe. C'est d'ailleurs le travail de cette première partie de stage : valider une réutilisation de la méthode d'extraction du cycle de rapport minimal sur des images satellites, à but d'extraire des régions correspondant aux zones urbaines.

# Chapitre 2

# Etat de l'art

Dans le travail d'extraction des zones urbaines d'une image, il y a deux parties bien distinctes. En premier lieu, il faut prétraiter les données image dont on dispose pour mettre en valeur les éléments sur lequels on souhaite s'attarder. Il faut trouver une méthode pour faire ressortir de l'image les pixels correspondant à de la ville, et diminuer l'importance de tous les autres. Bien évidement, cela repose sur un a priori des caractéristiques d'une ville. Ensuite, une fois qu'on a prétraité l'image, il faut extriare la région, en segmentant. On utilise les résultats de la description urbaines, et on essaie de séparer les zones réellement urbaines des autres zones.

## 2.1 Description urbaine

Les principales méthodes d'extraction de zones urbaines sont basées sur l'analyse de texture. L'analyse de texture est très difficile car les textures réelles ne sont souvent pas uniformes, à cause des changements d'orientation et de taille. De plus, beacuoup d'algorithmes proposés sont d'une grande complexité algorithmique. On remarquera que l'on utilise souvent des champs de Markov, pour leur capacité à modeliser des interactions parfois lointaines.

On peut trouver des états de l'art dans Haralick et Shapiro (1992), Chellappa et al. (1993), Reed et Du Buf (1993). Tuceryan et Jain (1993) ont divisé l'analyse de texture en quatre catégories :
- statistique
- geometrique
- basé modèle
- analyse du signal.

Les méthodes statistiques analysent la distribution dans l'espace des niveaux de gris, en calculant des caractéristiques locales en chaque point de l'image, puis en calculant des statistiques de ces paramètres locaux. Des méthodes de premier, deuxième ordre, ainsi que d'ordre supérieur existent, et permettent de calculer moyenne, variance et ressemblance de structures. Les plus utilisées sont les paramètres de cooccurrence (Haralick 73) et les différences de niveau de gris. (Weszka et al. 1976). Des méthodes basées sur l'autocorrélation ont aussi été développées, mais pour des résultats décevants. (Conners et Harlow 1980).

Les méthodes géométriques décomposent les textures en primitives, pour ensuite deviner des règles sur leur organisation spatiale. Ces primitives peuvent être extraites par détection de contours (Tuceryan et Jain 1990), ou par morphologie mathématique (Serra 1982). L'organisation des structures est alors apprise par des méthodes statistiques, mais aussi par diagrammes de Voronoï (Tuceryan et Jain 1990).

Les méthodes basées modèles font l'hypothèse de l'existence d'un modèle sous-jacent, régit par différents paramètres. On considère l'intensité des pixels comme fonction de deux éléments : un bruit additif aléatoire, et la structure de la surface de l'objet sur lequel repose la texture. On se réferrera à (Chellappa et al. 1993) pour de plus amples renseignements. On trouve des modèles ou l'éméent de base est la région, et d'autres où l'élément de base est le pixel; Mandelbrot (1983) a proposé un modèle basé fractal.

Les méthodes basées sur l'analyse du signal analysent le contenu fréquenciel de l'image. Il existe des filtres de domaine spatial (mask de Laws 1980), mais les masques standards (Robert et Sobel) peuvent aussi être utilisés pour obtenir des informations de fréquence. On peut aussi extraire de

l'information des moments (Tuceryan 1992). L'utilisation de la transformée de Fourier par fenêtre permet de garder de l'information sur la position mais on peut aussi utiliser des transformée en ondelettes (Mallat 1989). Lorsqu'on utilise des fonctions (somme de gaussiennes), on crée des bancs de filtres. Pour couvrir tout le spectre sans faire exploser la dimensionalité, (Manjunath et Ma 1996) en améliorant la conception des filtres. On pourra se référer à Reed and Wechsler (1990 ) and Reed and Wechsler (1991) pour une plus longue description sur ces méthodes d'analyse du signal.

Dans [14], il est proposé différents paramètres de texture pour l'analyse et l'identification de zones urbaines dans les images aériennes. Une étude complète sur une zone urbaine, ainsi qu'une étude intra-urbaine sont présentées.

La société SCOT[1], en accord avec Eurosat[2], propose dans sa gamme de logiciels des outils pour la délimitation d'agglomérations urbaines par le traitement automatique d'images satellitaires. Ils recherchent des logiciels et des algorithmes appropriés pour l'extraction de l'urbain, et proposent des méthodologies permettant de faire des tests sur différentes villes, avec différentes images.

Anne Lorette présente dans sa thèse [11] une méthode pour déterminer la position des villes sur des images satellites, en construisant des champs de Markov pour analyser la texture en chaque pixel, puis en utilisant un K-means flou pour classifier ses résultats. Sa méthode est presque entièrement automatisée, seule une étape de choix des classes réellement urbaine doit être réalisée par un opérateur.

## 2.2 Segmentation

Il existe de nombreuses méthodes pour segmenter une image. Entre autres, les méthodes par "level sets" donnent des résultats très interessants. Cependant, nous ne nous interessons pour l'instant qu'à une méthode basée sur la segmentation de graphe. On essayera des méthodes variationneles (voir le dernier chapitre (5)) dans la deuxième partie du stage.

Dans un article de 1993[18], Wu et Leahy proposaient une méthode pour segmenter un graphe en deux masses distinctes, en se basant sur la notion de "minimum cut", c'est-à-dire en choisissant la segmentation qui minimise le coût de séparation du graphe en deux agglomérats. Ce critère peut être utilisé pour obtenir de bons résultats de segmentation sur quelques images. Cependant, cette méthode a tendance à favoriser les petits ensembles de noeuds du graphe. Elle est donc biaisée vers les petites régions, ce qui n'est pas souhaitable.

Shi et Malik[16] ont poussé plus loin l'analyse en proposant de normaliser les régions par la mesure de la similarité entre régions, c'est à dire qu'il y a une certaine mesure de la taille de l'agrégat qui est prise en compte. On obtient ainsi une approche qui n'est plus biaisée vers de petites régions. Cependant, on ne peut pas inclure des information sur la frontière de la région considérée, car les régions sont vues comme des ensembles de points, et la spécificité des points sur le bord n'est pas exploitable en tant que telle. De plus, ils ne proposent qu'une solution approchée au problème qu'ils se posent. En effet, la résolution du "normalized cut" est un problème très dur (NP-Complet).

Cox et al. [2] présentent une approche de rapport d'énergie, entre une aire généralisée et une longueur généralisée, et trouvent la courbe optimal en choisissant chaque point de l'image comme pivot, l'un après l'autre. Cependant, ils sont restreint aux aires généralisée, ie l'intégrale de fonctions positives sur la région. De plus, l'algorithme est très lourd à implémenter.

Utilisant une autre approche, Jermyn et Ishikawa [7], qui se basent sur les travaux de Lawler [9] et de Meggido [13], proposent de résoudre la minimisation d'un rapport d'énergie sur un graphe. Ils proposent une solution exacte et en **temps polynomial** au problème qu'ils se sont posé. De plus, dans le même article, ils se servent d'une méthode donnée par Karp [8] pour appliquer une solution hautement parallélisable et multirésolution au problème du rapport moyen minimum sur un cycle dans un graphe, appliqué sur des images. On est cependant limités dans les fonctions qui peuvent être utilisées pour décrire les régions, par exemple il n'est pas directement possible d'introduire de l'information du deuxième ordre. La détection de régions à trou, par exemple un tore, n'est pas possible, car dans leur formalisme une région est décrite par l'intérieur d'un cycle.

[1] Service et conception de systèmes en observation de la Terre : `http://www.scot-sa.com`
[2] `http://www.eurostat.org`

De plus, la méthode n'est pas interactive, et ne permet pas à l'utilisateur de spécifier des points de recherche initiaux, du fait du minimum global. Cependant ces extensions peuvent être envisagée, en élaguant le graphe des régions non désirées.

L'article de Wang et Siskind[17] présente une nouvelle fonction, "ratio-cut", pour partitionner un graphe, ainsi qu'un algorithme (limité aux graphes planaires connexes) qui résoud le problème en temps polynomial, en se basant sur les travaux de Jermyn et Ishikawa [7] dans une de leurs étapes de réduction. De plus, ils étendent leur méthode afin de s'affranchir de bruit poivre-et-sel, et de bords flous, en construisant une heuristique. Ils accélèrent aussi la méthode en travaillant sur des sous-images qui sont ensuite fusionnées.

# Chapitre 3

# Eléments théoriques

## 3.1 Théorie des graphes

Deux excellentes références sur la théorie des graphes sont le Gondran et Minoux [6], et le livre de Ahuja et al. [1]. Après les définitions, je présenterais les algorithmes standards que j'utilise, puis je présenterais la méthode utilisée par Jermyn et Ishikawa dans [7], méthode que je reprends dans mon travail de stage.

### 3.1.1 Définitions

- Un graph orienté $G = (S, A)$ est défini comme un ensemble S, dont les éléments sont appelés les sommets, et un ensemble A, ensemble ordonné de couples de sommets, appelés arêtes.

- Un chemin de longueur $l$ est une suite de $l$ arcs noté $C = \{a_1, \ldots, a_l\}$ avec $a_1 = (s_0, s_1), a_2 = (s_1, s_2), \ldots, a_l = (s_{l-1}, s_l)$. Un chemin peut être vu comme le parcours d'une suite de sommets en prenant les arcs du graphe.

- Un cycle est un chemin tel que $s_0 = s_l$.

- On peut construire une fonction poids $w : A \rightarrow E$ qui associe à chaque arête une valeur, par exemple un réel ou un couple d'entiers.

### 3.1.2 Algorithmes standards

Je présente ici quelques problèmes classiques en théorie des graphes, et des algorithmes qui les résolvent. Je les utilise lorsqu'il faut extraire un cycle de poids négatif d'un graphe. Je commence par le classique algorithme de Dijkstra, qui trouve la distance entre un sommet particulier nommé source, et tous les autres sommets. Ensuite, une version adaptée aux poids négatifs est présentée (correction d'étiquettes) et finalement je montrerai comment trouver un cycle négatif, s'il existe, dans un graphe de poids quelconques.

*Plus court chemin*
*Soit $G$ un graphe pondéré, $w : A \rightarrow \mathbb{R}^+$ une fonction de poids positive. Trouver le plus court chemin entre un sommet source $s$ et tous les autres sommets.*

```
DIJKSTRA()
 1  T := ∅;        T̄ := S;
 2  d(i) := ∞ pour chaque sommet i ∈ S;
 3  d(s) := 0;       pred(s) := ∅;
 4  tant que T̄ ≠ ∅ faire
 5        soit i ∈ T̄ tel que d(i) = min{d(j)|j ∈ T̄}
 6        T := T ∪ {i}
 7        T̄ := T̄ − {i}
 8        pour  toute arête (i, j) faire
 9              si d(j) > d(i) + w_ij alors
10                 d(j) := d(i) + w_ij
11                 pred(j) := i;
```

Le principe est de garder deux ensembles de sommets, $T$ et $\bar{T}$. Tous les sommets dans $T$ ont déjà leur distance finale. On choisit alors dans l'ensemble $\bar{T}$ le sommet dont la distance est la plus petite. Ce noeud est à sa distance minimale. On l'insère dans l'ensemble $T$, on met à jour ses successeurs et on itère. Les distances sont bien calculées à partir du sommet source $s$, car il est le seul à n'être pas infini à l'initialisation, c'est avec celui-là que les calculs de distance commencent. L'algorithme termine avec tous les sommets ayant un attribut $d$ qui correspond à la distance de ce sommet à la source $s$.

L'algorithme de Dijkstra résoud le problème du *plus court chemin*, mais ne marche que pour des poids positifs car l'hypothèse qui fait le socle de l'algorithme est que tous les noeuds de $T$ ont déjà leur distance à la source finale. Or si l'on a un arc de poids négatif, on risque de modifier la distance d'un des noeuds de $T$. Dans le cas de poids négatifs, on risque d'avoir un cycle de poids négatif dans le graphe. En le parcourant, on mettrait tout le temps à jour les noeuds dont la distance serait indéfiniment décroissante, vers $-\infty$. Il existe un algorithme pour le cas où les arêtes peuvent être négatives, mais où on garantit qu'il n'y a pas de cycles négatifs.

```
CORRECTION D'ETIQUETTES()
 1  d(s) := 0;        pred(s) := 0;
 2  d(i) := ∞ pour chaque sommet i ∈ S − {s};
 3  LIST := {s};
 4  tant que LIST ≠ ∅ faire
 5       enlever un sommet i ∈ LIST
 6       pour  toute arête (i, j) faire
 7            si d(j) > d(i) + w_ij alors
 8               d(j) := d(i) + w_ij
 9               pred(j) := i;
10               si j ∉ LIST alors
11                  ajouter noeud j à LIST
```

On choisit un sommet de la liste. On met à jour tous les successeurs de ce sommet. On met dans la liste les successeurs, car leurs fils ne sont peut-être plus à jour, il faudra donc les réactualiser.

Finalement, on a les outils pour résoudre la

*Détection de cycles négatifs dans un graphe*
*On se donne $G$ un graphe pondéré par $w : A \to \mathbb{R}$, une fonction de poids quelconque. Trouver un cycle négatif, s'il en existe un.*

On a plusieurs façons de résoudre le problème, en se basant sur la mise à jour des distances des noeuds. Pour la simplicité de l'exposé, on supposera que le graphe est accessible à partir de la racine, ie pour tout sommet $x$, il existe un chemin de la source à $x$ (dans le cas contraire, il faut chercher indépendament dans chaque partie connexe).

- *Borne inférieure :* si le graphe ne contient pas de cycles négatifs, alors la distance de la source à tout noeud est supérieure à $|S| * Min$, où $Min = \min(\min\ w, 0)$ $(\star)$. On fait donc tourner un algorithme de correction d'étiquette, en choisissant arbitrairement le noeud source, et on attend un des évènements suivants :
  - la condition $(\star)$ est bafouée, un cycle négatif passe par le sommet de poids minimal.
  - l'algorithme termine, alors il n'y a pas de cycle négatif.
- *Recherche de cycle négatif :* On se contente de vérifier de temps en temps que le graphe des prédécesseurs ($pred(y) = x \Leftrightarrow d(y) = d(x) + w_{x,y}$) n'a pas de cycle, qui correspondrait alors à un cycle négatif dans le graphe original. Cette recherche peut être faite en "colorant" le graphe des prédecesseurs, la rencontre de la couleur déjà existante sur un noeud à colorier représentant un cycle.

### 3.1.3 Algorithme rapport de coût

Dans [7], Jermyn et Ishikawa appliquent à des images une méthode basée sur les travaux de Lawler [9] et de Meggido [13]. Ils y expliquent comment extraire le cycle de rapport minimal sur un bi-graphe $(G, A, S, (\lambda, \tau))$ dont les arêtes possèdent deux poids $\lambda$ et $\tau$. Le principe est de plonger la recherche de région dans un graphe. On se donne une image initiale. On construit alors une projection de l'image sur les bi-graphes, on résoud le problème du rapport minimal sur les cycles

de ce graphe, puis on retourne dans l'espace des images, où le cycle apparaît alors comme la frontière d'une région. Le grand intérêt est que l'algorithme sur les graphes ne propose qu'un seul résultat, le minimum **global** sur tous les poids de cycles du graphe, le poids d'un cycle $C$ étant défini par

$$w(C) = \frac{\sum_{a \in C} \lambda}{\sum_{a \in C} \tau} \tag{3.1}$$

L'algorithme est général, dans le sens qu'il n'a pas de paramétrisation liée aux données sur lequel il travaille. La contrepartie est qu'on doit par contre trouver la fonction de projection des images, ie une méthode de construction d'un graphe (sommets et arêtes) qui reflète les données de l'image, et qui souligne les zones qui répondent au critère que l'on recherche. L'idée est que le cycle qui sera extrait est composé d'arêtes dont la somme a une forte valeur. La modélisation a donc lieu lors de la construction de ce graphe, c'est là que la description de ce qu'on cherche est introduite.

On cherche à trouver la solution générale au problème suivant :

*Cycle de poids minimal*
*Soit un bi-graphe* $(G, S, A, (\lambda, \tau))$ *vérifiant*
- $\lambda : A \to \mathbb{Z}$,
- $\tau : A \to \mathbb{N}^\star$,
- *Si* $(u, (\lambda, \tau), v)$ *est une arête de* $G$, *alors* $(v, (-\lambda, \tau), u)$ *en est une autre.*

*Trouver le cycle de poids* $w = \frac{\sum \lambda}{\sum \tau}$ *minimal.*

On se restreint aux cycles qui ne font pas plusieurs boucles sur eux-mêmes, car ils ont le même rapport que le cycle qui ne fait qu'un seul tour. La solution ne peut pas être un cycle qui se croise, car sinon une des deux parties de cycles possède un rapport meilleur que la boucle. Cependant on peut trouver des cas dégénérés où plusieurs cycles ont le même poids.

L'algorithme fut d'abord décrit par Lawler [9], puis généralisé par Meggido [13]. Il repose sur l'observation suivante : soit $w_t : A \to \mathbb{Q},\ w_t(a) = \lambda(a) - t\tau(a),\ t \in \mathbb{Q}$. Si $t^*$ est le plus grand t tel l'équation $w_{t^*}(C) < 0$ n'admet pas de solution sur l'espace des cycles de $G$, alors $t^*$ est la valeur de $t$ telle la solution au problème du *Cycle de poids minimal* est le cycle $C$ qui vérifie $w(C) = t^*$.

On a donc réduit le problème à trouver $t^*$, ie la valeur de $t$ telle que le cycle de poids minimum, en considérant les poids $w_t$, est zero. Ceci revient alors à chercher la plus grande valeur (négative) de $t$ telle que le graphe $(G, w_t)$ n'a pas de cycle négatif. On remarquera bien que le graphe $(G, w_t)$ a une fonction poids qui envoie les arêtes sur les rationels, et non plus sur un couple d'entiers.

RechercheCycleMin()
1 $t = 0$ // borne supérieure sur t
2 **tant que** (ilExisteUnCycleNegatif$(G, t)$) **faire**
3 $t =$ rapport d'un cycle négatif de$G$
4 **rendre** trouverCycleDePoidsNul$(G, t)$

Concernant l'initialisation, on remarque que si les cycles ne sont pas tous de poids $w$ nul, il existe toujours un cycle de poids négatif. En effet, si $C$ est un cycle de poids $w(C) > 0$, alors $-C$, le même cycle, mais en prenant les arêtes à l'envers, est de poids négatif : $w(-C) < 0$. Il faut bien voir aussi que dans l'algorithme, le rapport des cycles n'est utilisé que pour mettre $t$ à jour. Le poids d'un cycle lors de la phase de recherche de cycles négatifs est donné par $w_t(C) = \sum(\lambda(a) - t\tau(a))$

Il faut remarquer que bien que le résultat cherché soit un cycle C, dont le poids est un rapport $w(c) = \frac{\sum \lambda}{\sum \tau}$, que l'on cherche à minimiser, l'algorithme travaille sur des poids $w_t(C) = \sum \lambda - t\tau$. Juste avant que l'algorithme ne rende son résultat, la valeur t est le plus petit rapport de poids des cycles de $G$. Il suffit alors d'extraire ce cycle du graphe. On peut le faire en considérant le graphe $(\tilde{G}, A, S, \tilde{w})$ où $\tilde{w}(u, v) = w(u, v) - d(v) + d(u)$. Dans ce graphe, les arêtes de poids non nul correspondent à des arêtes qui ne sont pas sur le chemin menant à un cycle. On peut donc les supprimer, et rechercher un cycle dans le graphe beaucoup moins dense (toutes les arêtes sont alors de poids nul) qui reste.

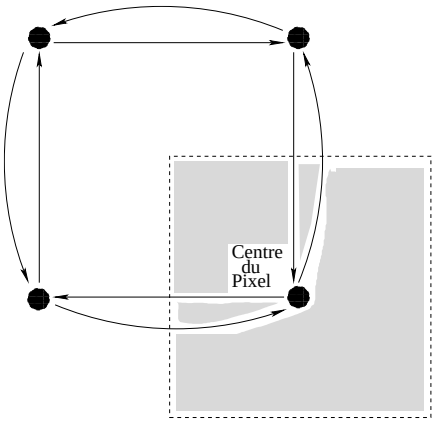
On remarquera que de toutes manières, la valeur de $t$ décroît jusqu'à atteindre $t^*$. On peut donc utiliser une dichotomie. On a une valeur d'où commencer, car 0 est une borne supérieure. On peut donc commencer avec $t = -1$ et multiplier cette valeur tant qu'il y a des cycles négatifs dans le graphe. Lorsqu'on n'en trouve plus, $t$ est plus grand que $t^*$. Par dichotomie, on se rapproche alors de $t^*$ en comsidérant qu'un cycle négatif trouvé signifie que $t^*$ est plus petit, et pas de cycle négatif signifie que $t^*$ est plus grand. J'ai décidé d'arrêter la dichotomie lorsque l'intervalle de recherche est plus petit qu'une constante que l'on choisi (1 est une bonne valeur). On réutilise alors l'algorithme linéaire décrit ci-dessus pour trouver la valeur exacte de $t^*$, ce que ne peut pas faire la dichotomie (considérer le cas où $t^* = -\frac{5}{3}$, qui ne peut pas être atteint par une dichotomie, qui ne peut atteindre que des fractions de dénominateur en $2^k$). L'avantage de la dichotomie est que, en un nombre raisonnable d'itérations, on se trouve proche de la valeur finale de $t^*$, ce qui n'est pas garanti par la version linéaire, qui pourrait essayer *tous* les cycles du graphe avant de trouver le bon. Cependant, la dichotomie teste souvent des valeurs plus petites que $t^*$. Il faut alors mener l'algorithme de correction d'étiquettes à son terme, ce qui est très couteux, car il faut trouver la distance de chaque nœud à la source. Cependant, l'expérience montre que la dichotomie est plus rapide.

### 3.1.4 Construction du graphe

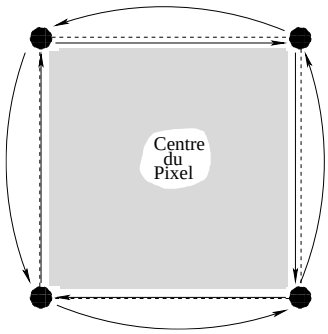
On est capable, par l'algorithme décrit dans la section précédente, d'extraire le cycle de rapport minimal d'un bi-graphe. Cependant, la construction d'un tel graphe, et les raisons qui poussent certaines arêtes à appartenir au cycle élu n'ont pas été explicitées. Je montrerai dans cette partie comment placer les sommets du graphe, pour correspondre aux pixels, ainsi que comment calculer le poids de chaque arête.

#### Localisation des sommets

On souhaite obtenir une correspondance entre l'image initiale et un cycle dans le graphe. Pour la localisation spatiale des sommets, deux solutions se présentent (voir figure 3.1). Les deux versions sont implémentées, mais la version où les sommets du graphe sont les angles des pixels s'explique mieux intuitivement, et permet de mieux définir le poids des arêtes du graphe en fonction de l'image de départ.




On peut placer les sommets du graphe au centre des pixels. Les arêtes représentent alors les relations de voisinage entre pixels. Cheminer le long d'un arête revient à changer de pixel. Les régions obtenues seront alors composées des pixels sur le cycle, ainsi que des pixels à l'intérieur.




On peut placer les sommets du graphe dans les coins des pixels. Les arêtes correspondent alors au bord des régions, qui sont constituées des pixels à l'intérieur des arêtes du cycle.

FIG. 3.1 – Différents positionnement des sommets du graphe.

**Poids des arêtes**

On va construire les arêtes telles que l'algorithme cherche la région $R$ qui minimise l'énergie

$$E[\partial R] = \frac{N[\partial R]}{D[\partial R]} = \frac{\int_S \gamma'(t)^\perp \cdot v(\gamma(t))\mathrm{d}t}{\int_S |\gamma'(t)| g(\gamma(t))\mathrm{d}t} \qquad (1) \tag{3.2}$$

avec
- $\partial R$ la frontière de la region,
- $S = S^1$ un cercle,
- $\gamma : S \to \mathbb{R}^2$ , qui représente la frontière de la région $\delta R$dans $\mathbb{R}^2$
- $v : \mathbb{R}^2 \to \mathbb{R}^2$ un champ de vecteurs,
- $g : \mathbb{R}^2 \to \mathbb{R}$.

Cette énergie peut paraître étonnante au premier coup d'œil : mise sous cette forme, c'est l'expression d'une énergie d'une région dans le domaine continu. En fait, comme on souhaite rester le plus général possible, l'expression sur le plan est plus intéressante que celle sur un domaine discret. La forme finale de l'énergie (fonctions $v$ et $g$) est à choisir en fonction d'à priori sur la région que l'on souhaite extraire, et doivent dépendre des données initiales. Ensuite, on discrétise cette énergie pour pouvoir la faire passer dans l'algorithme décrit plus haut. Cependant, je vais proposer une réécriture de l'énergie pour une manipulation plus aisée.

On conserve le dénominateur sous cette forme pour les informations sur la frontière. Donc pour calculer les valeurs de $\tau$ pour chaque arête, il faut intégrer la fonction g entre les deux sommets, sur une ligne droite. On calcule donc pour chaque arête $a : I \to \mathbb{R}^2$

$$\tau = \int_I g(a(s))\mathrm{d}s \tag{3.3}$$

Par exemple, on peut prendre $g = 1$. On a alors $\tau \in \{1, \sqrt{2}\}$, si l'on est en 8-connexité. De plus, si l'on regarde alors la somme de ces valeurs sur les cycles, on trouve que l'on a une petite valeur pour des petites régions. En fait, pour $g = 1$, on mesure le périmètre. Cette énergie a tendance à créer des régions convexes et pousse la courbe à être moins tordue. En effet, beaucoup de circonvolutions feront augmenter la valeur du dénominateur, tandis que la ligne droite produira le dénominateur le plus petit.

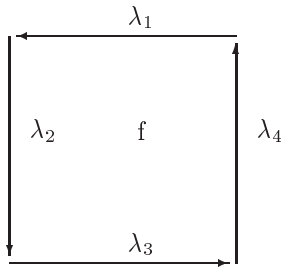
Par contre, grâce au théorème de Green,

$$\int_S \gamma'^\perp(t) \cdot v(\gamma(t))dt = \pm \int_R \nabla \cdot v(x, y)dxdy$$

(le signe dépend de l'orientation de la région), on peut transformer une intégrale sur un contour en une intégrale sur une région. En prenant donc une fonction $f = \nabla \cdot v$ (1), on peut apporter des informations sur la région entière dans le numérateur, et on remarquera que le champ vectoriel $v : \mathbb{R}^2 \to \mathbb{R}^2$

$$v_f^x(x, y) = \int_0^x f(x', y)\mathrm{d}x'$$
$$v_f^y(x, y) = 0$$

permet de retrouver la relation (1). Cette forme est très élégante car ce champ de vecteurs n'a qu'une coordonnée significative, ce qui permet de mettre à zéro toutes les arêtes horizontales.




On construit alors les poids $\lambda$ des arêtes par

$$\lambda_4 = -\lambda_2 + f \tag{3.4}$$

$$\lambda_3 = \lambda_1 = 0$$

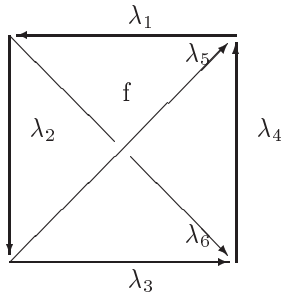
en n'oubliant pas de construire les symétriques :

$$(u, (\lambda, \tau), v) \in G \Longrightarrow (v, (-\lambda, \tau), u) \in G$$

On remarque que la somme des poids $\lambda$ sur le cycle correspond à la somme des valeurs de f dans la région ,

$$\sum_{cycle} \lambda = \sum_{region} f \tag{3.5}$$

C'est l'analogue du théorème de Green sur les graphes. En effet, on a construit les valeurs $\lambda$ des arêtes (en intégrant dans la relation (3.4)) pour que cette relation soit conséquence de la construction. On peut aussi construire des diagonales (8-connexité), en ajoutant la valeur de la moitié du pixel traversé, ce qui conserve la propriété (3.5), si l'orientation est respectée.




On met come valeurs

$$\lambda_5 = -\lambda_2 + \frac{f}{2}$$
$$\lambda_6 = \lambda_2 - \frac{f}{2}$$

Pour résumer, on peut dire qu'on est parti d'un rapport d'intégrales autour d'une région dans le domaine continu, que l'on a discrétisé sur un graphe par le quotient de sommes sur des arêtes :

$$E[R] = \frac{\int_R A(x)\mathrm{d}x}{\int_{\partial R} B(\gamma(s))\mathrm{d}s} \approx \frac{\sum_{a \in cycle} \lambda(a)}{\sum_{a \in cycle} \tau(a)} \tag{3.6}$$

## 3.2 Champs de Markov

On se rappelle que le but final est d'extraire des zones urbaines d'images satellites. A l'aide des algorithmes présentés dans la section ci-dessus, on sait trouver dans un graphe le cycle de rapport minimal, qui correspond à la région minimum global du critère utilisé. On a aussi dit que pour passer d'une image à un graphe, il fallait construire une fonction qui sache prendre en compte la modélisation des régions que l'on souhaite trouver.

On présente donc ici des travaux d'Anne Lorette sur les images de télédétection, qui décrivent une méthode utilisant des champs de Markov pour caractériser les pixels selon un indice de texture. Cet indice caractérise assez bien les zones urbaines, dans le sens qu'une valeur élevée de l'indice correspond très probablement à un pixel dans une ville, et un indice de valeur basse est synonyme de texture appartenant à autre chose qu'une ville. Cela va nous permettre d'avoir une première «impression» de la localisation des zones urbaines, qui sera alors segmentée par l'algorithme de rapport de coûts.

Dans la première section, je décris brièvement les caractéristiques d'un champ de Markov, qui sont nécessaires pour présenter les travaux d'Anne Lorette centrés sur la détection de zones urbaines.

### 3.2.1 Définitions

On renvoie à l'ouvrage de Li [10] pour une présentation générale des champs de Markov et leur utilisation en vision par ordinateur, ainsi qu'à [15] pour le lien entre les champs de Markov et la théorie des graphes.

L'image est représentée comme un ensemble de sites S.

Définition :
V est un voisinage du site $s$ si

1. $s \notin V(s)$
2. $s \in V(t) \Leftrightarrow t \in V(s)$

On considère le champ aléatoire $X = (X_1 \dots X_n)$, défini sur $S$, à image dans $\Omega$.

Définition :
$(X, \Omega, P)$ est un champ de Markov par rapport au système de voisinage V s'il vérifie :

1. $\forall x \in \Omega, P(X = x) > 0$
2. $\forall s \in S,\ \forall x \in \Omega,\ P(X_s = x_s | X_r = x_r, r \in S - \{s\}) = P(X_s = x_s | X_r = x_r, r \in V(s))$

Cela signifie que la valeur d'un site du champ ne dépend que de la valeur de ses voisins. Cependant, on peut quand même introduire des proporiétés globales en imposant des contraintes locales.

Théorème d'Hammersley-Clifford :
Si $(X, \Omega, P)$ est un champ de Markov, alors sa distribution $P(X)$ est une distribution de Gibbs :

$$P(X) = \frac{e^{-U(X)}}{Z} \quad \text{où } Z = \sum_{X \in \Omega} e^{-U(X)} \text{ et } U(X) = \sum_{c \in C} V_c(X_s, s \in c)$$

Pour être plus exhaustif, il faudrait présenter quelques modèles (Ising, Potts, ...), un exemple d'utilisation de champ de Markov (restauration), et les estimateurs habituels (MAP, MPM...). Ensuite la relaxation probabiliste avec Métropolis et l'echantilloneur de Gibbs, le principe du recuit simulé et quelques approximations, genre ICM.

### 3.2.2 Indice de texture pour les zones urbaines

Dans sa thèse [11], (on pourra aussi se référer à l'article [12]) Anne Lorette, en se basant sur des travaux de Xavier Descombes [3], décrit une méthode permettant d'extraire les zones urbaines dans une image satellite (données provenant d'une simulation SPOT5), basée sur l'estimation d'un paramètre appelé température. Ensuite, elle segmente ce résultat, et présente aussi des résultats sur de la classification intra-urbaine. Cependant, je vais uniquement parler de la méthode pour décrire les régions urbaines, et je ne parlerai pas de la segmentation, car je ne m'en sers pas.

Dans un premier temps je décrirai le modèle abstrait utilisé. Ensuite, je présenterai la méthode d'estimation des paramètres et pour terminer, je donnerai l'utilisation que Anne Lorette en fait pour extraire son paramètre final d'indice urbain.

#### Le modèle

Le paramètre de base que l'on souhaite extraire, la température, notée $T$, est fondé sur un modèle markovien. Le but est de construire les modèles tels que les zones urbaines qui sont des zones à forte variance et peu corrélées aient une température plus élevée que les zones de champs.

On définit la probabilité conditionnelle pour le modèle sur un voisinage composé de deux voisins par

$$\begin{aligned} P(X_s \mid X_r \in V(s)) &= P(X_s \mid m_s) \\ &= \frac{1}{Z_{V(s)}} \exp -\beta \left( \sum_{r \in V(s)} (X_s - X_r)^2 + \lambda (X_s - \mu)^2 \right) \\ &\equiv N\left( \frac{2m_s + \mu\lambda}{2 + \lambda}, \frac{1}{2\beta(2+\lambda)} \right) \end{aligned}$$

où $m_s$ désigne la moyenne des deux voisins du site $s$, et $\beta = \frac{1}{T}$ et $\lambda$ sont les paramètres du modèle. La probabilité conditionnelle suit une loi normale. Le paramètre que l'on va extraire est la variance conditionnelle :

$$\sigma^2 = \frac{1}{2\beta(2+\lambda)}$$

**Estimation**

Il s'agit d'estimer la variance $\sigma^2$.

Anne Lorette estime les paramètres de texture, les variances conditionnelles locales, en utilisant la méthode des «queues de comètes». Cette méthode, fondée sur l'estimation des matrices des probabilités conditionnelles, a été proposée pour la première fois par Xavier Descombes dans [3], puis détaillée dans [4, 5]. La méthode consiste à estimer la matrice des probabilités conditionnelles dans une fenêtre glissante, centrée sur le pixel $X_s$. Il faut que la taille de la fenêtre soit assez grande pour avoir des résultats fiables, et assez petite pour éviter les mélanges de texture. Pour chaque position de la fenêtre (donc pour chaque pixel), on estime la variance conditionnelle locale à partir de la matrice des probabilités conditionnelles. Par exemple, dans la direction (NOo/SEe), la probabilité conditionnelle locale est :

$$P\left(X_s | X_r \in V_s^{(d=NOo/See)}\right) = P\left(X_s | \frac{1}{2}(X(NOo) + X(SEe))\right)$$

Elle dispose ainsi d'un modèle qu'elle sait estimer. Elle propose alors de le construire sur un voisinage un peu plus étendu que d'habitude : elle choisit de prendre huit directions différentes pour chaque pixel, et de construire pour chaque direction un estimateur. Le voisinage $V_s^{(d)}$ d'un pixel $s$ reste composé de ses deux plus proches voisins dans la direction considérée, un devant, un derrière. Le choix d'un voisinage non standard (ni 4 ni 8 connexe) est une originalité intéressante.

**Paramètre final**

On rappelle que le but final est de trouver un indice de texture répondant fortement aux zones urbaines. Pour cela, Anne Lorette a choisi de construire huit températures pour chaque pixel, dans huit directions différentes. Cependant, comme les voisinages ne sont pas isotropes, elle introduit aussi une étape de normalisation, qui permet de comparer les huit valeurs entre elles.

L'hypothèse de base est que la zone urbaine répond fortement au modèle présenté ci-dessus dans toutes les directions. Cependant, prendre le max des valeurs, ou le min, ne donne pas de résultat satisfaisant. En fait, les huit valeurs extraites étant plus ou moins grandes, et sensibles à d'autres éléments que la ville, elle a proposé de ne garder comme valeur finale que la moyenne des valeurs médianes, ie $\frac{1}{2}(p_4 + p_5)$, où les valeurs $p_i, i \in 1 \dots 8$ sont les paramètres extraits, triés par ordre croissant de valeur. Elle obtient ainsi une carte de valeurs, où il n'y a qu'un indice de texture urbaine pour chaque pixel.

Les images de valeurs d'Anne Lorette représentent une fonction qui à chaque pixel associe un réel positif. L'image de cette fonction est un candidat possible à l'entrée de l'algorithme de rapport de coût sur les graphes. En combinant les deux méthodes, on espère trouver les différentes régions de l'image originale, qui devraient correspondre à des villes. On tirera profit du fait que la réponse est construite pour que l'intensité la plus forte soit située sur les zones urbaines.

# Chapitre 4

# Application

On rappelle que l'on utilise l'algorithme de rapport de coûts pour extraire des régions d'une image, la nouveauté résidant dans le choix de fonctions pour décrire les comportements que l'on souhaite voir ressortir de l'image.

On souhaite choisir deux fonctions $f$ et $g$ telles que $g$ décrive le poids $\tau$, et $f$ soit intégrée sur les ligne horizontales, de sorte que lambda décrive la somme des valeurs avant l'arête. On se réferera à la section 3.1.4 ainsi qu'aux équations 3.3 et 3.4 pour la méthode employée.

Lorsque qu'une l'arête se trouve sur le contour d'une région, on souhaite qu'elle appartienne au cycle qui sera finalement extrait. Il faut donc construire la fonction $g$ telle qu'elle ait des petites valeurs sur un contour, tandis que la fonction $f$ aura des grandes valeurs dans la région, de telle sorte que la somme de la fonction $v$ sur le cycle prenne une grande valeur.

J'ai d'abord testé sur des images de synthèse, pour valider mon implémentation de l'algorithme de coût minimum, avant de m'attaquer à l'intégration des résultats d'Anne Lorette, pour avoir comme résultat l'extraction de zones urbaines.

## 4.1 Application à des images synthétiques

Il est facile de projetter des images directement sur une grille : on construit un graphe dont les sommets sont les coins des pixels, et les arêtes les bords des pixels. On intègre la valeur du niveau de gris des pixels pour construire la fonction $\lambda$, tandis que la fonction $\tau$ est la distance entre deux noeuds (en 8-connexité, $\tau \in \{1, \sqrt{2}\}$).

On obtient de bons résultats en prenant $g = 1$ et $f = |\nabla I|$, le gradient de l'intensité de l'image synthétique $I$ de départ, car cela correspond à attribuer une forte valeur aux contours. La région à l'intérieur des contours est donc choisie car inclure les bords augmente la valeur absolue de l'énergie, tandis qu'aller au-delà augmente la longueur sans ajouter au numérateur, ce qui a pour résultat de faire diminuer la valeur absolue.

On constate que la méthode marche bien pour extraire les contours, ce qui est le résultat attendu (voir figure (4.1)).

De plus, on voit que le modèle est bien adapté aux régions en longueur. Par exemple, les queues ne sont pas coupées, et font partie de la région extraite de l'image 4.2

Même lorsque les contours ne sont pas très clairs, l'algorithme prend quand même la partie importante de la région : le résultat visuel est satisfaisant (figure 4.3).

## 4.2 Application à des images de télédétection

### 4.2.1 Données brutes

Si l'on entre directement dans l'algorithme les valeurs des variances conditionnelles données par la méthode d'Anne Lorette (que l'on notera $\Phi$), la région qui est invariablement trouvée est l'image entière. En effet, les valeurs des pixels sont trop grandes par rapport à la longueur d'une arête du graphe. Comme le modèle initial, qui ne comporte pas de paramètre, ne permet pas de trouver de

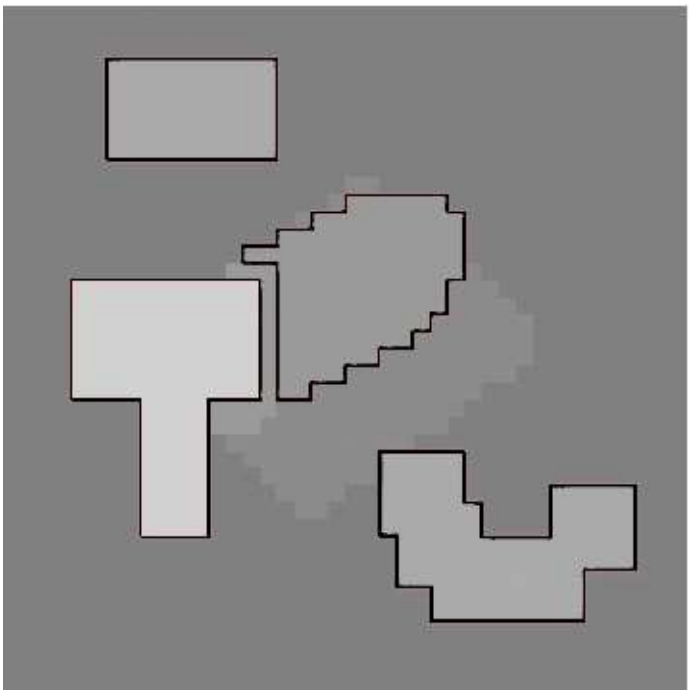


FIG. 4.1 – Extraction de régions

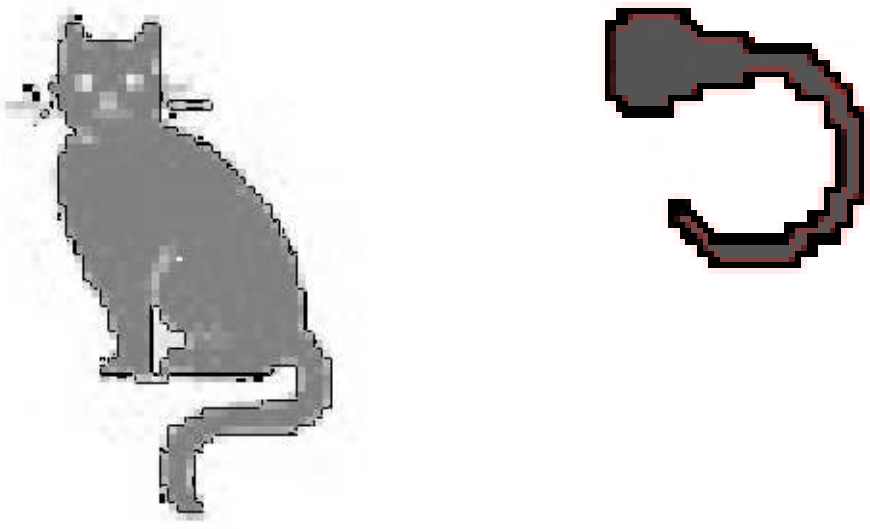


FIG. 4.2 – Extraction des régions, mêmes fines (voir les queues)

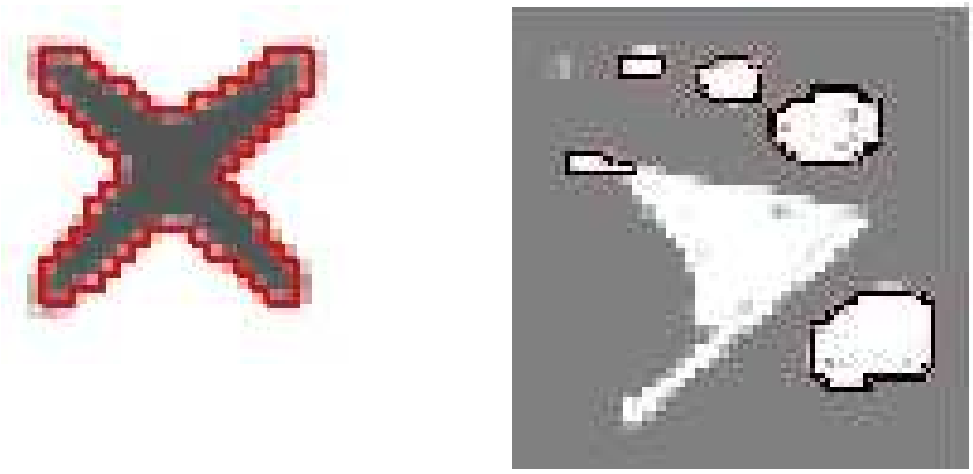


FIG. 4.3 – Extraction des régions, bords flous

région dans les données fournies, on est amenés à modifier la fonction de projection sur le graphe. On introduit des modèles plus compliqués, par exemple incluant le gradient, pour espérer extraire les zones urbaines. Les premiers essais ont porté sur la soustraction d'une constante, ce qui revient à recentrer les données, et sur une modification du dénominateur, en donnant à chaque arête une longueur dépendant du gradient sous-jacent.

### 4.2.2 Ajout d'une constante

On se rappele que le modèle est invariant à la multiplication par un scalaire, mais pas à l'addition : les résultats obtenus avec la fonction $f$ ne sont pas forcément les mêmes que ceux obtenus avec $f + \alpha$, pour $\alpha \in R$. La première idée pour obtenir des resultats sur la sortie d'Anne Lorette est donc de rajouter une constante. On obtient effectivement des résultats différents, avec le meilleur à l'oeil nu étant $f = \Phi - 450$. Mais cette valeur dépend de l'image, et une méthode pour déterminer ce paramètre devrait être envisagée pour utiliser cette méthode. Cette fonction trop simpliste a donc été rejettée, car bien qu'elle présente une avancée par rapport aux données brutes, elle donnait des résultats encore trop loin de ce qu'on était en droit d'espérer, avec en plus l'ajout d'un paramètre.

Je présente quelques résultats (voir Figure 4.4) pour l'ajout d'une constante, car ils sont intéressants du point de vue du comportement de l'algorithme. Lorsqu'on ajoute 0 à la fonction, on se retrouve avec toute la région. On essaie donc avec d'autres valeurs, plus petites. On remarque que pour $\alpha < -400$ on commence à trouver une région. Il semble y avoir une région minimum vers $-450$, puis la région recommence à grandir, et finit par englober à nouveau toute l'image. En fait, comme l'ajout d'une constante recentre les données, on voit qu'il y a une "meilleure" valeur qui correspond à recentrer les valeurs, de telle manière que la contribution des points «ville» soit réellement plus importante que celle des points extra-urbains. En effet, comme la valeur des points extra-urbain n'est pas forcément nulle, ils contribuent quand même aux régions, et les prendre en compte n'est pas mauvais au regard de l'algorithme, sauf si justement on recentre les données. On remarque qu'ensuite enlever une trop grande valeur produit l'effet mirroir. Les valeurs que l'on ne souhaite pas voir sont à nouveau sollicitées, car en changeant l'orientation de la région, la valeur est grande à nouveau pour des pixels extra-urbains.

### 4.2.3 Gradient de l'image

En utilisant non pas la valeur de $\Phi$ brute, mais le gradient de l'image ($f = |\nabla\Phi|$), on se retrouve avec des petites zones, car certaines régions (généralement au centre des villes) répondent très fortement. On n'obtient pas de plus grandes régions car la méthode n'est pas biaisée vers les régions de grande taille. On se retrouve plutôt avec les pics de la fonction $\Phi$, qui sont entourés de forts gradients dans toutes les directions. Cette méthode n'est donc pas satisfaisante, car on ne recherche pas les centres des zones urbaines, mais les zones entières, en soulignant la frontière avec le non urbain (voir figure 4.5).

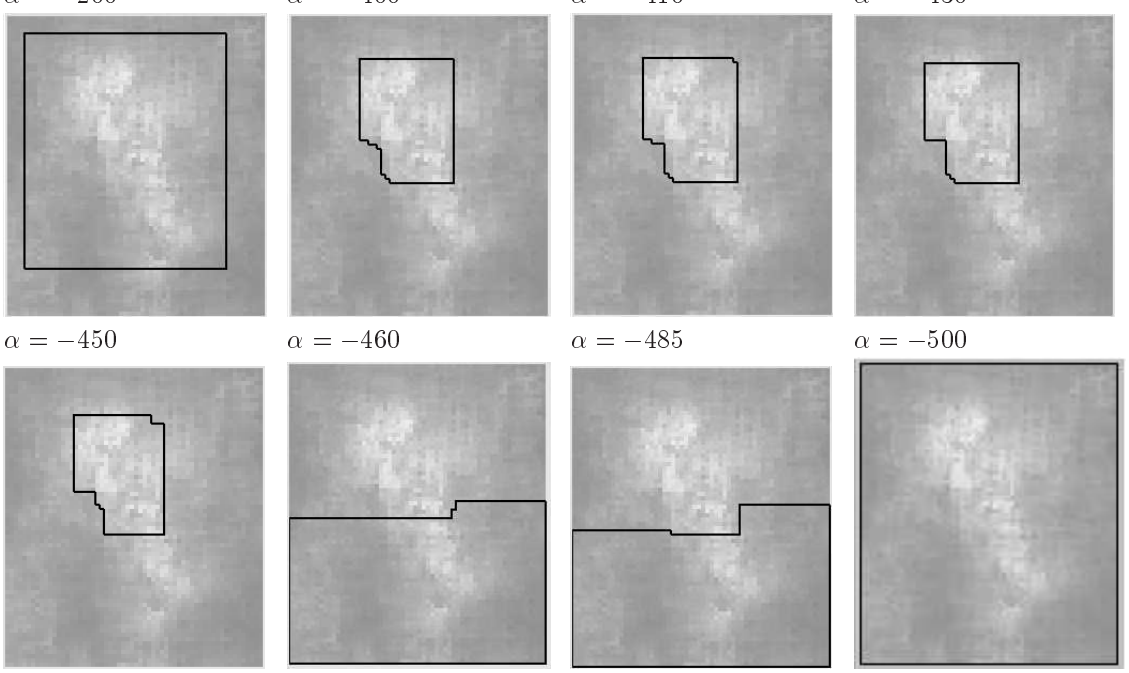



FIG. 4.4 – Différents valeurs de $\alpha$, où $f = \Phi + \alpha$

### 4.2.4 Longueur d'une arête en fonction du gradient

On souhaite avoir un poids d'arête important sur le contour. On essaie d'inclure cette contrainte dans le dénominateur. La fonction $g = 1$ agit comme un terme d'élasticité, en ressérant la courbe, mais ne tient pas compte du tout de l'image sous-jacente. Comme on cherche les bords de régions, et que sur le contour d'une région, le gradient a une valeur importante, on a choisi pour la fonction du dénominateur

$$g = \frac{1}{\beta + |\nabla I|} \qquad \begin{array}{l} \text{si } |\nabla I| \ll 1 \text{ alors } g \sim 1 \text{ et } \frac{\lambda}{\tau} \sim \lambda \\ \text{si } |\nabla I| \gg 1 \text{ alors } g \ll 1 \text{ et } \frac{\lambda}{\tau} \gg 1 \end{array}$$

La valeur de $\beta$ change en fait la contribution relative d'un grand gradient par rapport à un petit gradient. Pour $\beta$ petit (proche de 1 par exemple), il y aura une grande variation dans les valeurs de $\tau$, tandis que pour $\beta$ égal au gradient moyen, on n'aura qu'une variation du simple au triple. Cependant, on se retrouve avec à nouveau des paramètres à regler. Je ne présente pas de résultats pour cette méthode, car je n'ai pas encore eu le temps de bien regarder quels étaient les réglages qui convenaient le mieux.

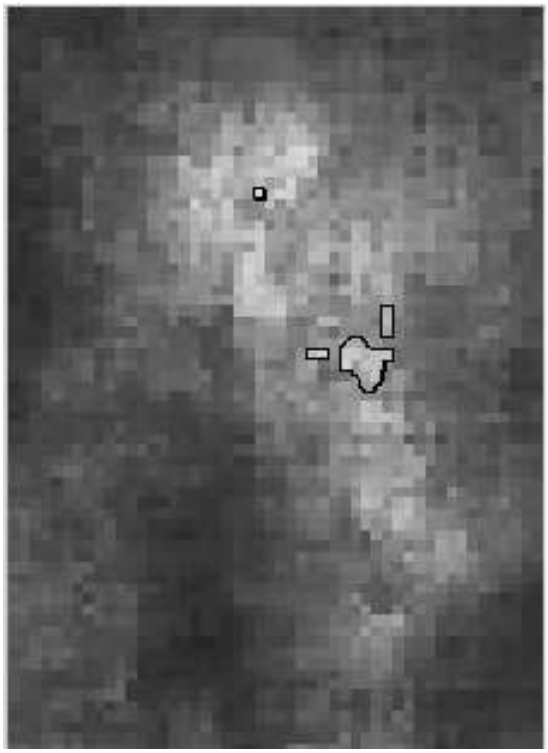


FIG. 4.5 – Une partie d'une image satellite, où on utilise uniquement le gradient de la fonction $\Phi$ pour extraire les zones urbaines

# Chapitre 5

# Conclusion et perspectives

Nous avons ici implémenté l'algorithme de Jermyn-Ishikawa et avons exhibé des fonctions candidats potentiels à l'application à l'extraction de zones urbaines sur des images satellites.

L'algorithme garantissant un mimimum global, on est certain d'avoir la région optimale pour le critère considéré, au contraire de méthodes comme la descente de gradient, où le résultat final est un minimum local, sans aucun contrôle sur la distance au minimum global. Les fonctions peuvent être vues comme les paramètres du modèles, les données étant le prétraitement d'une image satellite brute, qui a été repris sur les résultats du travail de thèse d'Anne Lorette. On se heurte alors à la difficulté de modéliser proprement les propriétés que l'on souhaite voir ressortir de l'image.

Dans la suite de ce stage, on s'intéressera à introduire la notion de rapport d'énergie dans les méthodes variationnelles standards (level sets). La méthode habituelle consiste à dériver l'énergie, pour obtenir les équations d'évolution :

$$E(\gamma) = \int F[\gamma, I]$$

$$\frac{\delta E}{\delta \gamma(t)} = G[\gamma(t), \gamma'(t), \gamma''(t), I]$$

La variation d'énergie ne dépend habituellement que d'éléments locaux comme la valeurs des quelques pixels sous-jacents, de la tangente à la courbe, ou du rayon de courbure.

Par contre, dans le cadre de

$$E(\gamma) = \frac{E_1(\gamma)}{E_2(\gamma} \ ,$$

la variation d'énergie devient

$$\delta E = \frac{\delta E_1(\gamma)}{E_2(\gamma)} - \frac{E_1(\gamma)}{E_2^2(\gamma)} \delta E_2(\gamma)$$

On retrouve dans l'équation d'évolution les intégrales $E_1$ et $E_2$. Cela signifie que l'évolution n'est plus uniquement liée à des phénomènes locaux comme le rayon de courbure, mais aussi à l'ensemble de la région que la courbe englobe.

Dans le cadre des intégrales sur toute la région, M. Barlaud a proposé des énergies tenant compte de l'intérieur de la région (moyenne). Ce genre d'information pourrait être inclu dans le numérateur d'une énergie.

# Bibliographie


[1] Ahuja, R., Magnanti, T., and Orlin, J. *Network flows.* Prentice Hall, 1993.

[2] Cox, I. J., Rao, S. B., and Zhong, Y. Ratio regions : A technique for image segmentation. *Proceedings 13th International Conference on Pattern Recognition* (1996), 557–564.

[3] Descombes, X. *Champs markoviens an analyse d'image.* Thèse de doctorat, ENST, December 1993.

[4] Descombes, X., and Prêteux, F. Topology and parameter estimation in markov random field modeling. *SPIE, Neural and Stochastic Methods in Image and Signal Processing II* (1993), 156–166.

[5] Descombes, X., Sigelle, M., and Prêteux, F. Estimating gaussian markov random field parameters in a non-stationary framework : Application to remote sensing imaging. *IEEE Trans. on Image Processing 8*, 4 (1999), 490–503.

[6] Gondran, M., and Minoux, M. *Graphs and Algorithms.* Wiley interscience, 1988.

[7] Jermyn, I., and Ishikawa, H. Globally optimal regions and boudaries as minimum ratio weight cycles. *IEEE Trans Pattern Analysis and Machine Intelligence 23*, 10 (2001), 1075–1088.

[8] Karp, R. A characterization of the minimum cycle mean in a digraph. *Discrete Math. 23* (1978), 93–112.

[9] Lawler, E. L. Optimal cycles in doubly weighted linear graphs. *Proceedings of International Sympsosium on Theory of Graphs* (1966), 209–213.

[10] Li, S. *Markov Random Field modeling in computer vision.* Springer-Verlag, 1995.

[11] Lorette, A. *Analyse de texture par méthodes markoviennes et par morphologie mathématique : application à l'analyse des zones urbaines sur des images satellitales.* Thèse de doctorat, Université de Nice - Sophia Antipolis, 1999.

[12] Lorette, A., Descombes, X., and Zerubia, J. Texture analysis through a markovian modelling and fuzzy classification : Application to urban area extraction from satellite images. *Internation Journal of Computer Vision 3*, 36 (2000), 221–236.

[13] Meggido, N. Combinatorial optimization with rational objective function. *Math. Operations Research 4* (1979), 414–424.

[14] Paquerault, S. Traitement d'images aériennes : Reconaissance des zones urbaines sur des critères de texture. *rapport de DEA, ENST Paris* (1994).

[15] Pérez, P. *Modèles et algorithmes pour l'analyse probabiliste des images.* Habilitation à diriger les recherches, Université de Rennes 1, December 2003.

[16] Shi, J., and Malik, J. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* (2000).

[17] Wang, S., and Siskind, J. M. Image segmentation with ratio cut - extended version.

[18] Wu, Z., and Leahy, R. An optimal graph theoretic approach to data clustering : Theory and its application to image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell. 15*, 11 (1993), 1101–1113.