Méthodes de segmentation d'images à base de graphes

1.1 Introduction

Un reproche qui peut être fait aux algorithmes traditionnels de traitement d'images est le manque de flexibilité des structures de données impliquées. L'organisation sous forme de simple grille génère en effet d'importantes contraintes vis-à-vis de leur exécution, limitant leur potentiel. Cela s'illustre particulièrement pour des problématiques liées à la « forme » des données. Par exemple, de nombreux algorithmes sont – sur le principe – capables de traiter des données de plus grande dimension, telles que des images tri-dimensionnelles ou encore des vidéos. Cependant, si un tel algorithme n'a pas été dès l'origine implémenté de façon à supporter des données de plus haute dimension, il est relativement complexe de le faire évoluer dans ce sens, et des développements spécifiques sont souvent requis [1].

De par leur structure, les graphes sont capables de représenter tout domaine discret. Il est ainsi possible de construire une représentation d'une image à l'aide d'un graphe en associant un nœud du graphe à chaque pixel. Par la suite, en établissant des relations entre les nœuds correspondants à des pixels voisins dans l'image, il est possible d'obtenir un graphe dont la topologie reflète l'organisation initiale des pixels, notamment les relations de voisinage.

Cette représentation est beaucoup plus souple. Étant donné qu'il n'y a plus de notion de « bord de l'image », exécuter un algorithme sur une région d'intérêt – qu'elle soit rectangulaire ou non – peut simplement revenir à travailler sur un sous-graphe composé uniquement des nœuds appartenant à cette région. Par ailleurs, la structure de graphe faisant abstraction des dimensions de l'image, il est envisageable de traiter des données bi et tridimensionnelles exactement de la même manière ; seule l'étape de construction du graphe nécessite d'être adaptée aux données manipulées.

Comme nous le verrons plus loin dans ce chapitre, de nombreux outils de traitement d'images basés sur les graphes reposent sur des algorithmes issus d'autre domaines, princi-

1. Nous pouvons arriver à une conclusion similaire en considérant la problématique de traitement de région d'intérêt de formes arbitraires, en particulier non rectangulaires.

palement de la théorie des graphes. Il devient alors possible de concevoir des algorithmes de traitement d'images à partir d'approches conçues pour des problématiques totalement différentes. Réciproquement, les graphes rendent possible d'appliquer des algorithmes de traitement d'images à des données qui ne sont pas des images. Ainsi, un algorithme de débruitage peut, par exemple, servir à lisser un maillage [Elmoataz *et al.*, 2008].

Ce chapitre constitue un état de l'art des méthodes de traitement d'images à base de graphes. Nous réaliserons tout d'abord un rappel de notions usuelles de la théorie des graphes, puis enchaînerons sur les différentes méthodes permettant de représenter une image à l'aide d'un graphe. Nous terminerons ce chapitre par une revue des différentes familles de méthodes de segmentation d'images basées sur les graphes.

1.2 Généralités sur les graphes, définitions et notations

Un graphe est une représentation d'un ensemble d'éléments dans laquelle des relations peuvent être établies entre toutes paires d'éléments. Ceux-ci sont représentés par des nœuds, et les relations par des arêtes. Un graphe $G = (V, E)$ est ainsi composé d'un ensemble $V = \{V_1 \dots V_N\}$ de N nœuds et d'un ensemble $E \subset V \times V$ d'arêtes. Dans ce manuscrit, nous ne nous intéressons qu'aux graphes simples (il n'existe pas d'arête reliant un nœud à lui-même et il y a au plus une arête reliant deux nœuds) et non-orientés (si une relation existe entre deux nœuds, celle-ci est symétrique).

Deux nœuds $u, v \in V$ reliés par une arête $(u, v) \in E$ sont dit adjacents ou encore voisins. Cette relation est notée $u \sim v$. Réciproquement, l'arête (u, v) est dite incidente aux nœuds u et v.

Le voisinage $\mathcal{N}(u)$ d'un nœud u correspond quant à lui à l'ensemble des nœuds adjacents à u :

$$\mathcal{N}(u) = \{v \in V \text{ tel que } (u, v) \in E\}$$

On note $\mathcal{H}(V)$ l'espace de Hilbert des fonctions scalaires à valeurs réelles $f : V \to \mathbb{R}$ définies sur l'ensemble des nœuds et $\mathcal{H}(E)$ l'espace de Hilbert des fonctions scalaires à valeurs réelles $w : E \to \mathbb{R}$ définies sur l'ensemble des arêtes. Ces ensembles sont munis des produits scalaires suivants :

$$\langle f, h \rangle_{\mathcal{H}(V)} = \sum_{u \in V} f(u) \cdot h(u) \quad \text{avec } f, h : V \to \mathbb{R} \tag{1.1}$$

$$\langle F, H \rangle_{\mathcal{H}(E)} = \sum_{(u,v) \in E} F(u, v) \cdot H(u, v) \quad \text{avec } F, H : E \to \mathbb{R} \tag{1.2}$$

Une fonction $w \in \mathcal{H}(E)$ associe à chaque arête $(u, v) \in E$ une valeur réelle $w(u, v) \in \mathbb{R}$ (aussi notée w_{uv}). Une telle fonction est généralement appelée fonction de pondération, et permet de définir un graphe pondéré noté $G = (V, E, w)$.

Nous noterons par ailleurs $\mathcal{H}^m(V)$ l'espace de Hilbert des fonctions vectorielles à valeurs réelles $f : V \to \mathbb{R}^m$ définies sur l'ensemble des nœuds. Ces fonctions sont généralement utilisées pour représenter les données associées à chaque nœud, sous forme de vecteur d'attributs. La notation indicée $f_i(\cdot)$, $i \in \{1 \dots m\}$ permet d'accéder à chacune des composantes du vecteur.

1.3 Graphes de similarité

La plupart des méthodes de segmentation d'images à base de graphes utilisent des graphes de similarité. Leur rôle est de permettre la représentation de relations entre les composantes de l'image possédant des caractéristiques semblables, la fonction de pondération servant alors à moduler ces relations en représentant un « degré de similarité ».

Ces composantes peuvent être de différents types. Comme évoqué précédemment, avec une approche bas niveau, chaque nœud représente un pixel de l'image. Nous verrons plus loin qu'il est également possible de considérer des éléments de plus haut niveau, tels que des régions de l'image, permettant ainsi de diminuer la taille du graphe et d'accélérer son traitement.

Le point commun entre les différents types de composantes est qu'il leur est associé un ou plusieurs attributs les caractérisant, représentés par des fonctions de $\mathcal{H}(V)$ ou $\mathcal{H}^m(V)$. Ceux-ci correspondent aux données à traiter et peuvent, là encore, être de types très différents : coordonnées [Bougleux *et al.*, 2007], intensité du niveau de gris [Felzenszwalb et Huttenlocher, 2004, Grady, 2006], caractéristiques colorimétriques [Shi et Malik, 2000, Lezoray *et al.*, 2007], caractéristiques de textures [Shi et Malik, 2000], patchs d'images [Lezoray *et al.*, 2008]... Nous reviendrons dans un prochain chapitre sur les principales méthodes de caractérisation de textures.

La construction d'un graphe de similarité peut être découpée en trois étapes, que nous allons détailler dans les sections suivantes :
- la sélection de l'ensemble des nœuds,
- la sélection de l'ensemble des arêtes,
- la sélection de la mesure de similarité.

1.3.1 Sélection de l'ensemble des nœuds

En ce qui concerne la sélection de l'ensemble des nœuds, l'approche la plus naturelle consiste à représenter chaque donnée par un nœud du graphe [Chan *et al.*, 2001, Bougleux, 2007, Lezoray *et al.*, 2008]. L'intérêt est ici d'avoir une représentation la plus fine possible des interactions entre les données.

Malheureusement, le nombre d'arêtes potentielles augmentant de façon quadratique avec le nombre de nœuds, ce niveau de détails peut avoir un coût non négligeable. Cela est particulièrement vrai dans le cadre du traitement d'images où les données manipulées sont les pixels, dont le nombre peut être relativement important. Nous verrons dans la section suivante les solutions qui permettent de conserver les problèmes de segmentation solvables tout en maintenant un maximum d'informations utiles.

L'alternative consiste à réduire le nombre de nœuds en employant une approche de type « superpixel ». Cela revient à appliquer un algorithme de *clustering* comme prétraitement de l'image dans le but de produire une sur-segmentation de celle-ci. La partition ainsi obtenue est constituée de régions (ou superpixels) présentant une très faible variabilité intrarégion. Le but de cette étape de *clustering* n'étant pas d'anticiper sur la segmentation finale [2], cette propriété est extrêmement importante. En effet, c'est elle qui permet de

2. La capacité de détection des contours des objets aura tout de même une influence certaine sur la

regrouper des pixels dont la dissociation – du fait de leur très forte ressemblance – n'aurait fourni aucune information supplémentaire à l'algorithme de segmentation qui sera par la suite appliqué.

Cette approche a notamment été utilisée dans [Ta, 2009], où l'auteur utilise l'algorithme de partitions d'énergie [Arbeláez et Cohen, 2004] pour effectuer une sur-segmentation des images et créer ses graphes, réduisant ainsi le nombre de nœuds à moins de 3 % du nombre initial de pixels.

Le lecteur intéressé pourra se référer à [Achanta *et al.*, 2012] pour un état de l'art des méthodes de « superpixellisation ».

1.3.2 Sélection de l'ensemble des arêtes

Choisir l'ensemble des arêtes consiste à définir la topologie du graphe, en particulier par la spécification du voisinage de chaque nœud. Avant d'entrer dans le détail des méthodes existantes, il faut garder à l'esprit qu'il est difficile de prédire quelle topologie de graphe permettra d'obtenir les meilleurs résultats. Le choix des heuristiques sous-jacentes est dépendant de l'algorithme utilisé et des données à traiter [Maier *et al.*, 2008].

Les méthodes de construction que nous allons décrire se basent généralement sur une mesure de distance permettant d'évaluer la proximité de deux nœuds. Cette proximité n'est cependant pas nécessairement géométrique ; elle peut tout à fait se baser sur une notion de similarité des caractéristiques associées aux nœuds, ou encore combiner distance géométrique et caractéristique. De nouveau, il n'existe pas de règle permettant de définir la mesure la plus adaptée ; ce choix devra être fait en fonction de l'objectif recherché et des données à traiter.

Soient $A \subset \mathbb{R}^m$ et $p, q \in A$. Les trois mesures de distance les plus utilisées sont :

– la norme L_1, ou distance de Manhattan :

$$d(p,q) = \sum_{i=1}^{m} |p_i - q_i| \quad , \tag{1.3}$$

– la norme L_2, ou distance Euclidienne :

$$d(p,q) = \sum_{i=1}^{m} (p_i - q_i)^2 \quad , \tag{1.4}$$

– la norme L_∞, ou distance de Chebychev :

$$d(p,q) = \max_i |p_i - q_i| \quad . \tag{1.5}$$

Dans l'absolu, la majorité des algorithmes de segmentation par graphes travaille à partir de graphes complets, c'est à dire dans lesquels une arête existe entre chaque paire de nœuds. Ce type de graphe possède l'avantage de mettre en interaction l'ensemble des nœuds, mais constitue aussi le graphe de similarité le plus complexe qui puisse être construit.

qualité du résultat final. Il est donc important que l'algorithme de *clustering* considère correctement les caractéristiques des pixels.

Son exhaustivité rend cette technique inapplicable dès que le nombre de nœuds devient important, ce qui est souvent le cas pour les images.

En réalité, il est généralement inutile de recourir à un tel niveau de détails car deux pixels géométriquement proches ont plus de chances de partager des caractéristiques communes – et donc d'avoir une influence l'un sur l'autre – que deux pixels géométriquement opposés. Dans un graphe complet, la relation qui existe entre des pixels géométriquement distants sera généralement inhibée, car associée à un poids de faible valeur relative. Il est donc courant de ne pas mettre en relation deux nœuds si ceux-ci ne satisfont pas un certain critère de similarité. Cela amène à la notion de voisinage, et donc à la mesure de distance évoquée précédemment.

Les deux types de graphes de voisinage généralement utilisés sont le graphe de ε-voisinage et le graphe des k plus proches voisins [Felzenszwalb et Huttenlocher, 2004, Lezoray *et al.*, 2007, von Luxburg, 2007, Ta *et al.*, 2009].

Dans un graphe de ε-voisinage, deux nœuds ne sont mis en relation que si la distance les séparant ne dépasse pas un seuil préalablement fixé. Soit $G_\varepsilon = (V, E)$ un graphe de ε-voisinage, et $\mu : V \times V \to \mathbb{R}$ une mesure de distance. Le voisinage d'un nœud u est défini par :

$$\mathcal{N}_\varepsilon(u) = \{v \in V \setminus \{u\} \text{ tel que } \mu(u, v) \leq \varepsilon\} \quad . \tag{1.6}$$

Cette méthode permet notamment de reproduire les structures en 4 et 8-voisinage selon lesquelles les images sont traditionnellement traitées. En se basant sur les coordonnées des pixels et en fixant $\varepsilon = 1$, la distance de Manhattan permet d'obtenir un graphe grille en 4-voisinage et la distance de Chebychev, un graphe grille en 8-voisinage. Appliquées à des images 3D, ces mêmes définitions restent valables, et permettent de construire respectivement des 6 et 26-voisinages.

Dans un graphe des k plus proches voisins, le voisinage $\mathcal{N}_{kNN}(u)$ d'un nœud u est constitué des k nœuds les plus proches de u du point de vue de la mesure de distance μ utilisée. Cependant, telle qu'illustrée figure 1.1, cette heuristique produit un graphe orienté : un nœud v peut faire partie des k plus proches voisins d'un nœud u sans que u ne fasse partie du voisinage de v. Deux graphes non-orientés peuvent être dérivés à partir du graphe des k plus proches voisins [von Luxburg, 2007, Maier *et al.*, 2009] :

- le graphe mutuel des k plus proches voisins, dans lequel deux nœuds u et v sont connectés si $v \in N_{kNN}(u)$ **ou** si $u \in N_{kNN}(v)$. Chaque nœud possède alors au moins k voisins,
- le graphe symétrique des k plus proches voisins, dans lequel deux nœuds u et v sont connectés si $v \in N_{kNN}(u)$ **et** si $u \in N_{kNN}(v)$. Chaque nœud possède alors au plus k voisins.

La version mutuelle est généralement préférée car elle assure un certain degré de connectivité à chaque nœud [Elmoataz *et al.*, 2008, Ta, 2009].

Les graphes de ε-voisinage sont principalement utilisés lorsque les données font partie d'un domaine organisé – tel que les images – pour lequel la proximité géométrique reste un critère de similarité important. Les graphes des k plus proches voisins sont, quant à eux, plus régulièrement employés pour traiter des données issues de domaines non-organisés, pour lesquels il n'existe pas de notion de proximité géométrique.

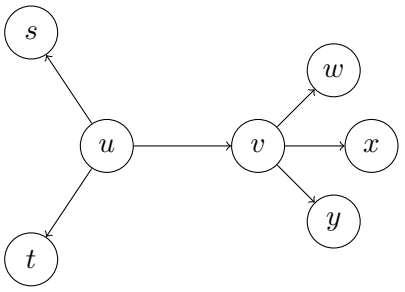

FIGURE 1.1 – Exemple de graphe orienté des 3 plus proches voisins (seuls les voisinages des nœuds u et v sont illustrés). La longueur des arêtes reflète la distance entre les nœuds. Le nœud v fait partie des 3 plus proches voisins du nœud u, mais le nœud u ne fait pas partie des 3 plus proches voisins du nœud v. Dans le graphe mutuel, le voisinage de u sera composé des nœuds s, t et v, tandis que celui de v sera composé de u, w, x et y. Dans le graphe symétrique, les nœuds u et v ne sont pas voisins.

Quelques travaux ont tenté d'évaluer l'influence du type de graphe, ou encore de la taille du voisinage. Dans [Felzenszwalb et Huttenlocher, 2004], le graphe des k plus proches voisins a été comparé au graphe de ε-voisinage. Un même algorithme de segmentation était alors exécuté dans ces deux configurations, et les performances comparées. Les résultats obtenus étaient relativement équivalents. Cependant, comme cela est confirmé dans [Grady et Jolly, 2008], l'utilisation de voisinages de taille trop réduite ne permet pas de tirer pleinement partie de l'information contenue dans les relations de voisinage. De fortes connectivités semblent donc être préférables, au détriment évident des performances de l'algorithme en matière de temps de calcul ainsi que d'occupation mémoire.

1.3.3 Sélection de la mesure de similarité

La similarité de deux nœuds est généralement inversement proportionnelle à la distance les séparant. À ce titre, il est tout à fait envisageable de pondérer la relation entre deux nœuds u et v à l'aide de la mesure de distance définie précédemment, en fixant par exemple $w_{uv} = (\mu(u,v) + \varepsilon)^{-1}$, avec ε un réel positif proche de zéro permettant d'éviter toute instabilité numérique.

Cependant, en particulier pour les domaines organisés tels que les images, une approche différente est généralement employée : la topologie du graphe est construite à l'aide d'heuristiques purement géométriques – permettant ainsi de limiter l'étendue des voisinages – tandis que la mesure de similarité est calculée à partir des données. Une mesure de distance, telle que celles présentées précédemment, est alors pondérée à l'aide d'une fonction de transfert décroissante, comme une fonction inverse :

$$w_{uv} = (\sigma \cdot \mu(u,v)^2 + \varepsilon)^{-1} \ , \tag{1.7}$$

ou encore une gaussienne :

$$w_{uv} = \exp\left(\frac{-\mu(u,v)^2}{\sigma}\right) \quad , \tag{1.8}$$

avec $\sigma > 0$ un paramètre dont le rôle est de régler la vitesse de décroissance de ces fonctions.

Il a également été suggéré d'utiliser des mesures anisotropiques, tenant à la fois compte de la proximité géométrique et de la similarité des caractéristiques :

$$w_{uv} = \frac{1}{d(u,v)} \cdot \frac{1}{\sigma \cdot \mu(u,v)^2 + \varepsilon} \quad , \tag{1.9}$$

$$w_{uv} = \frac{1}{d(u,v)} \cdot \exp\left(\frac{-\mu(u,v)^2}{\sigma}\right) \quad , \tag{1.10}$$

avec $d(u,v)$ une mesure de la distance géométrique entre les deux nœuds u et v. Les tests réalisés dans [Grady et Jolly, 2008] indiquent que la mesure de similarité basée sur la fonction inverse permet une meilleure pondération des relations.

Enfin, il faut noter que certains algorithmes peuvent se passer de mesure de similarité, la topologie du graphe fournissant déjà une information suffisante pour mener à bien la tâche associée. Il est alors utile de fixer l'ensemble des mesures de similarité à une valeur neutre et constante, permettant un traitement équitable de l'ensemble des voisins. Dans ce cas, $\forall\ (u,v) \in E, w_{uv} = 1$.

Ces trois étapes permettent de définir une structure de données utilisée par la plupart des méthodes de traitement d'images à base de graphes. Dans la suite de ce chapitre, nous allons détailler quatre familles de méthodes de segmentation.

1.4 Méthodes à base d'arbres de couverture minimale

En théorie des graphes, un graphe est un arbre s'il est simple, connexe et acyclique, c'est à dire s'il n'existe qu'un unique chemin entre chaque paire de nœuds.

Soit $G = (V, E, w)$ un graphe. On suppose que G est connexe (il existe un chemin entre toute paire de nœud). L'arbre de couverture minimale [3] de G correspond au graphe connexe $G' = (V, E', w)$ avec $E' \subset E$, tel que la somme des poids associés aux arêtes de l'ensemble E' est minimale. La figure 1.2 illustre cette notion.

Il faut noter que, à l'inverse de l'ensemble des algorithmes présentés dans ce chapitre, les algorithmes dont il est question dans cette section travaillent à partir de graphes de dissimilarité, dans lesquels la fonction de pondération mesure la « distance » entre les nœuds.

Dans un arbre, la suppression de k arêtes entraine la création de $k+1$ composantes connexe. Afin de réaliser le partitionnement d'un ensemble de données, l'idée générale est de supprimer les arêtes de l'arbre de couverture minimale présentant les poids les plus importants. Les composantes connexes résultante représentent alors le résultat de la segmentation. La figure 1.3 illustre ce principe.

3. Ou MST, de l'anglais *Minimum Spanning Tree*.

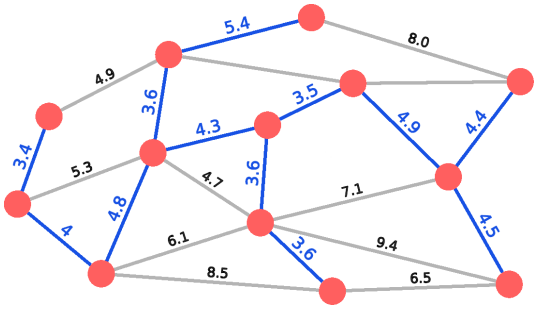

FIGURE 1.2 – Exemple d'arbre de couverture minimale. Le poids associé aux arêtes correspond à la distance Euclidienne entre chaque paire de nœuds. L'arbre de couverture minimale (en bleu) connecte l'ensemble des nœuds du graphe et minimise la somme des poids associés aux arêtes.

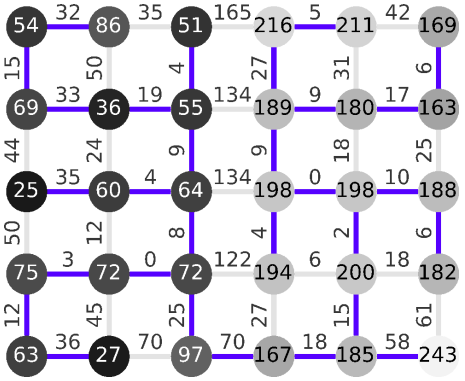

FIGURE 1.3 – Utilisation d'un arbre de couverture minimale pour la segmentation de données. La valeur associée à chaque nœud représente son niveau de gris, tandis que celle associée à chaque arête correspond à la différence de niveaux de gris (en valeur absolue) des nœuds qu'elle relie. Dans l'arbre de couverture minimale (représenté par les arêtes en bleu), l'arête de poid 70 (qui relie les deux régions au centre du graphe) est celle de poids maximal. Sa suppression entraine la création de deux composantes connexes, chacune identifiant une région du graphe dans laquelle les niveaux de gris sont homogènes.

[Zahn, 1971] a été le premier à appliquer ce concept au partitionnement de données. Dans un arbre de couverture minimale, il définit comme « inconsistante » une arête dont le poids associé est supérieur au poids moyen des arêtes des composantes connexes situées à ses extrémités. Il propose ainsi un algorithme de partitionnement dans lequel les arêtes inconsistantes sont supprimées du MST.

Les travaux de Zahn ont inspiré à [Felzenszwalb et Huttenlocher, 2004] un algorithme de segmentation d'images non-supervisé dont le principe s'inspire du plus célèbre algorithme de construction d'arbres de couverture minimale, l'algorithme de Kruskal [Kruskal, 1956].

L'algorithme de Kruskal considère deux graphes : $G = (V, E, w)$, qui représente les données initiales et $G_{MST} = (V, E_s, w)$ l'arbre de couverture minimale. Notons que l'ensemble de nœuds ainsi que la fonction de pondération de ces deux graphes sont identiques, et que $E_s \subset E$. À l'initialisation, $E_s = \emptyset$. Chaque arête de E est ensuite considérée par ordre croissant de poids. Si les composantes connexes de G situées aux deux extrémités d'une arête sont différentes, alors celle-ci est ajoutée au graphe G_{MST}. L'algorithme s'arrête lorsque toute les arêtes ont été considérées.

Une caractéristique importante de cet algorithme, qu'exploite Felzenszwalb et Huttenlocher, est que l'arbre de couverture minimale n'est pas construit à partir d'une unique entité, mais résulte de la fusion d'un ensemble de composantes connexes [4]. Étant donné que les arêtes sont considérées par ordre croissant de poids, chaque arête qui est ajoutée à l'arbre de couverture minimale joint deux nœuds qui sont plus « distants » que l'ensemble des paires de nœuds jusqu'alors considérées.

Felzenszwalb et Huttenlocher proposent donc de mesurer l'homogénéité d'une composante connexe à partir de sa « différence interne », définie comme égale au poids le plus important de son arbre de couverture minimale, et de conditionner la fusion de deux composantes à l'aide de cette mesure. Deux composantes ne sont ainsi fusionnées que si cela n'entraine pas la création d'une composante moins homogène, c'est à dire si il n'existe aucune arête entre ces deux composantes dont le poids serait supérieur à la différence interne de chacune des composantes.

Une version hiérarchique de cette approche est également proposée dans [Haxhimusa et Kropatsch, 2004].

1.5 Méthodes à base de coupes minimales et approches spectrales

Une coupe d'un graphe $G = (V, E)$ est définie comme étant la partition de celui-ci en deux sous-graphes disjoints S et T. Le coût – ou valeur – de cette coupe correspond au nombre d'arêtes joignant ces deux sous-graphes :

$$\text{cut}(S, T) = Card\{(u, v) \in E \text{ tel que } u \in S \text{ et } v \in T\} \quad . \tag{1.11}$$

Lorsqu'un graphe $G = (V, E, w)$ est pondéré, ce coût est alors égal à la somme des

4. À l'initialisation, chaque nœud défini une composante connexe.

poids des arêtes :

$$\mathrm{cut}(S,T) = \sum_{\substack{(u,v)\in E \\ u\in S \\ v\in T}} w_{uv} \quad . \tag{1.12}$$

À partir de cette notion, il est possible de définir le problème de calcul de coupe minimale : étant donnés un graphe $G = (V, E)$ et deux nœuds $s, t \in V$, trouver les régions S et T telles que $s \in S$, $t \in T$ et $S \cap T = \emptyset$ dont la coupe associée est de valeur minimale (la définition de ce problème est identique pour les graphes pondérés).

Lorsqu'il est associé à la notion de graphe de similarité, il est aisé de transformer ce problème de calcul de coupe minimale en un problème de segmentation d'images : étant donnés le graphe de similarité associé à une image et deux pixels représentatifs de deux régions à segmenter, calculer la coupe minimale séparant ces deux pixels permettra d'identifier les arêtes de plus faible poids, c'est à dire de « couper » entre les pixels présentant un minimum de similarité.

Cette approche a notamment été appliquée par Wu et Leahy [Wu et Leahy, 1993] qui emploient l'algorithme de Gomory-Hu [Gomory et Hu, 1961] afin de construire un arbre identifiant l'ensemble des coupes séparant toute paire de nœuds du graphe. La segmentation de l'image en K classes est alors réalisée en identifiant les $K - 1$ coupes de plus petites valeurs.

Malheureusement, l'algorithme de Gomory-Hu présente une complexité relativement importante ($\mathcal{O}(|V|^4)$, avec $|V|$ le nombre de nœuds du graphe), ce qui le rend inapplicable à de grands graphes. Partant du constat que la majorité des coupes identifiées n'est pas utilisée – car de valeur trop importante – Wu et Leahy ont proposé une implémentation hiérarchique de l'algorithme initial. Celle-ci vise à identifier des sous-graphes ne pouvant pas influer sur le calcul des coupes minimales afin de les condenser [5]. La taille du graphe sur lequel l'algorithme de Gomory-Hu est ensuite exécuté peut ainsi être réduite, permettant le traitement de graphes de grandes tailles. Les expérimentations alors menées ont permis de constater une amélioration notable du temps de calcul, celui-ci étant alors divisé par 72.

Cette approche est cependant biaisée. La valeur d'une coupe étant globalement proportionnelle au nombre d'arêtes « coupées », celles de très petites tailles sont favorisées, sans qu'elles n'aient de réelle signification du point de vue de la segmentation.

Afin de compenser ce biais, plusieurs travaux ont suggéré une mesure « normalisée » du coût d'une coupe, dans laquelle la taille de cette dernière est prise en compte.

Dans [Wang et Siskind, 2001], la notion de coupe moyenne est ainsi proposée. La normalisation du coût d'une coupe est réalisée en divisant le coût initial, tel que formulé par l'équation (1.12), par le nombre d'arêtes concernées. L'algorithme de calcul de coupe moyenne minimale – emprunté lui aussi à la théorie des graphes – se limite cependant à des graphes planaires [6] pour une résolution en temps polynomial, le problème étant NP-difficile pour des graphes de topologies arbitraires.

La méthode des coupes normalisées [Shi et Malik, 2000] est, quant à elle, radicalement

5. La condensation d'un sous-graphe revient à le représenter par un nœud unique.

6. Un graphe est qualifié de planaire s'il peut être représenté dans le plan sans qu'aucune arête n'en croise une autre.

différente. Le coût d'une coupe y est exprimé comme une proportion du poids des arêtes concernées par la coupe :

$$\mathrm{Ncut}(S,T) = \frac{\mathrm{cut}(S,T)}{\mathrm{cut}(S,V)} + \frac{\mathrm{cut}(S,T)}{\mathrm{cut}(T,V)} \quad , \tag{1.13}$$

avec V l'ensemble des nœuds. L'originalité de cette approche réside dans la méthode de résolution du problème de calcul de coupe minimale. Celle-ci fait en effet partie du domaine de la théorie spectrale des graphes, c'est à dire qu'elle se base sur une analyse des valeurs et vecteurs propres de la matrice Laplacienne du graphe, laquelle est définie par :

$$M_{uv} = \begin{cases} \sum_{s \in V} w_{us} & \text{si } u = v \\ -w_{uv} & \text{sinon} \end{cases} \qquad \forall u, v \in V \quad . \tag{1.14}$$

La propriété qui nous intéresse dans le cadre de la segmentation d'images est que le spectre du graphe, c'est à dire la liste des valeurs propres de sa matrice Laplacienne normalisée, est indicatif du nombre de composantes connexes du graphe. En particulier, si la matrice admet i valeurs propres nulles, cela signifie que le graphe est composé de $i+1$ composantes connexes, que les vecteurs propres associés permettent de caractériser. Cette propriété est étendue aux valeurs propres non-nulles : l'intensité d'une valeur propre est inversement proportionnelle à la connexité de la composante associée avec le reste du graphe. Ainsi, en considérant les i plus petites valeurs propres, il est possible d'identifier $i+1$ clusters de nœuds présentant une forte connexité interne et une faible connexité avec le reste du graphe.

Cette propriété peut être combinée à la formule du coût d'une coupe normalisée $\mathrm{Ncut}(S,T)$ sous la forme d'un problème aux valeurs propres généralisé. L'identification des $k-1$ plus petites valeurs propres issues de la solution du problème permet alors de segmenter l'image en k régions. La relaxation de certaines contraintes reste nécessaire afin de permettre la résolution du problème, celui-ci étant sinon NP-complet.

D'un point de vue purement algorithmique, la plus grosse difficulté de cette méthode revient à résoudre le problème aux valeurs propres généralisé, ce qui requiert actuellement $\mathcal{O}(n^3)$ opérations, avec n le nombre de nœuds du graphe. Il est évident que cette approche est impraticable, y compris pour des images de petite taille, ne serait-ce que pour stocker en mémoire la matrice Laplacienne. Heureusement, plusieurs propriétés permettent d'optimiser la résolution de cet algorithme. D'une part, en fonction de la méthode de construction du graphe, il se peut que la matrice d'adjacence (et donc la matrice Laplacienne) soit essentiellement composée de valeurs nulles [7]. Cette propriété peut être exploitée par de nombreuses implémentations de calcul matriciel afin d'optimiser l'espace mémoire occupé par ces matrices, mais aussi en réduisant le nombre d'opérations liées à la résolution du problème aux valeurs propres généralisé. D'autre part, le résultat de la segmentation n'est généré qu'à partir d'un nombre réduit de vecteurs propres, pour lesquels il n'est pas nécessaire d'avoir une grande précision. Ces deux dernières propriétés peuvent être exploitées par l'algorithme Lanczos [Golub et Van Loan, 1996], qui permet de réduire

7. On parle alors de matrice creuse.

la complexité du calcul des vecteurs propres en $\mathcal{O}\left(n^{3/2}\right)$, ce qui rend cette approche exploitable pour des images de taille modérée. Il reste cependant impossible de l'appliquer à des images de grande taille ou encore à des images 3D.

Malheureusement, la relaxation et les différentes approximations nécessaires à la résolution du problème rendent le résultat de la segmentation souvent difficile à interpréter ou à exploiter, comme illustré figure 1.4. Il est actuellement compliqué d'influer sur l'algorithme pour corriger ce genre de comportement car le seul élément sur lequel l'utilisateur a la possibilité intervenir est le graphe lui-même.

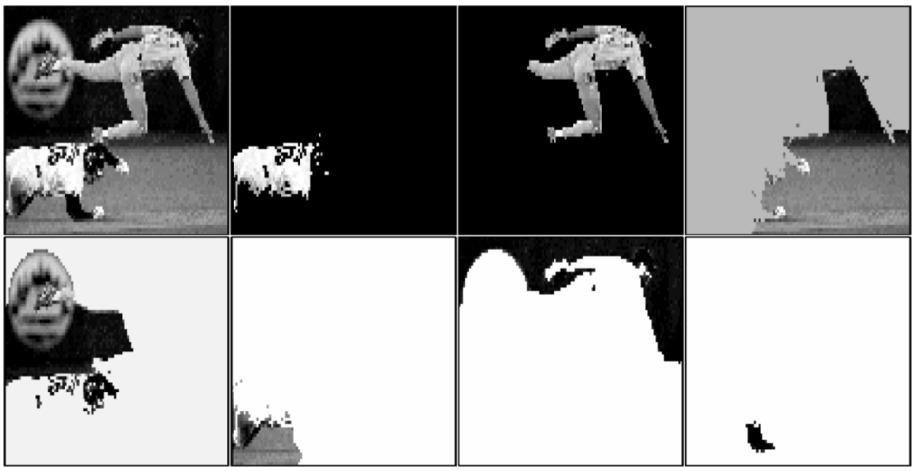

FIGURE 1.4 – Exemple de segmentation obtenue par la méthode des coupes normalisées minimales : l'image d'origine (en haut à gauche) et les partitions générées à partir des sept plus petites valeurs propres solutions du problème.

D'autres méthodes exploitant différentes matrices Laplaciennes ont également été explorées [Weiss, 1999, von Luxburg, 2007].

1.6 Marche aléatoire

La marche aléatoire correspond à la modélisation mathématique d'un processus dynamique discret – le « marcheur » – dont chaque évolution est définie à l'aide d'un processus stochastique de type chaîne de Markov : à chaque étape, le marcheur passe aléatoirement d'une position (ou état) à une autre selon une certaine probabilité qui ne dépend pas des positions passées mais uniquement de la position actuelle.

L'analyse du comportement d'un tel processus permet de mettre en évidence des relations entre différentes positions ou états auxquels le processus peut accéder. Un célèbre exemple d'application reposant sur ce principe est le robot d'indexation du moteur de recherche de Google, lequel a notamment pour tâche d'évaluer la popularité d'une page Web compte tenu de la popularité de celles pointant dessus.

Dans [Grady, 2006], l'auteur propose de traiter le problème de k-partition d'une image à l'aide d'un processus de marche aléatoire. Étant donnés un graphe de similarité et un ensemble de marqueurs manuellement positionnés, le problème revient à identifier sur quel

marqueur un marcheur aléatoire au départ d'un pixel a le plus de chance d'arriver en premier, la mesure de similarité du graphe étant interprétée comme une probabilité de déplacement.

La résolution de ce problème par la simulation d'un marcheur aléatoire est bien entendu inenvisageable. Cependant, il a été démontré que la probabilité qu'un marcheur aléatoire atteigne un marqueur correspond à la solution d'un problème de Dirichlet [Grady, 2006], qui consiste à calculer la fonction harmonique – c'est à dire dont la dérivée seconde est nulle en tout point – prolongeant les marqueurs. Ce problème peut être résolu à l'aide d'un système d'équations basé sur la matrice Laplacienne du graphe. Bien que la résolution d'un tel système d'équation soit relativement lourde et complexe, des optimisations tenant compte de la densité des matrices peuvent être appliquées.

1.7 Segmentation par régularisation de graphes

La régularisation d'un problème consiste à introduire dans sa formulation un ensemble de contraintes dans le but de le rendre soluble ou de prévenir l'obtention de solutions non-exploitables ou trop complexes. Cette technique est notamment utilisée dans le domaine de l'apprentissage automatique pour éviter le sur-apprentissage [Larsen et Hansen, 1994]. En traitement d'images, les contraintes utilisées concernent essentiellement la régularité de la solution, et permettent d'éviter une trop grande sensibilité au bruit.

Dans ses travaux, Ta propose d'aborder plusieurs problèmes de traitement d'images (colorisation, débruitage, segmentation) à l'aide d'une méthode de régularisation de graphes [Ta, 2009]. Pour cela, il s'appuie sur deux classes de modèles variationnels discrets qui permettent de transformer le problème initial en un problème de minimisation d'énergie. Celui-ci prend la forme suivante :

$$\min_{f \in \mathcal{H}(V)} \{E(f) = E_r(f) + \lambda \cdot E_d(f)\} \quad , \tag{1.15}$$

avec $E_d(f)$ le terme d'attache aux données lié à la pertinence de la solution f par rapport à l'objectif initial, $E_r(f)$ la fonctionnelle de régularisation qui vise à évaluer la régularité de la solution et λ le paramètre permettant d'exprimer un compromis entre la précision et la régularité de la solution.

Avant d'entrer dans le détail de ces modèles variationnels et d'en présenter les applications, il est nécessaire de définir la façon dont la fonctionnelle de régularisation évalue la régularité d'une fonction $f \in \mathcal{H}(V)$. Comme dans le domaine continu, la régularité d'une fonction peut être évaluée à l'aide d'opérateurs différentiels définis sur les graphes.

1.7.1 Opérateurs différentiels définis sur les graphes

Les travaux de Ta s'inspirent de méthodes variationnelles définies dans le domaine continu et basées sur les équations aux dérivées partielles. Ils reposent donc sur un ensemble d'opérateurs différentiels qui correspondent à des versions adaptées pour les graphes pondérés des opérateurs traditionnels d'analyse différentielle [Bougleux *et al.*, 2007,Elmoataz *et al.*, 2008].

Afin de présenter ces opérateurs, nous considérons un graphe pondéré $G = (V, E, w)$, auquel sont associées les fonctions scalaires à valeurs réelles $f \in \mathcal{H}(V)$ et $F \in \mathcal{H}(E)$.

1.7.1.1 Opérateur de différence pondérée et opérateur adjoint

Les deux opérateurs de base sont l'opérateur de différence pondérée et son adjoint [Bougleux *et al.*, 2007, Elmoataz *et al.*, 2008].

L'opérateur de différence pondérée correspond à l'opérateur de dérivée du domaine continu. Il s'exprime sous la forme d'une application $d_w : \mathcal{H}(V) \rightarrow \mathcal{H}(E)$ définie selon chaque arête uv par :

$$(d_w f)(u, v) = \frac{f(v) - f(u)}{h_{uv}} \quad , \tag{1.16}$$

avec h_{uv} le pas d'échantillonnage permettant de prendre en compte la « distance » entre u et v. Dans le cas des graphes, ce pas permet de faire intervenir la mesure de similarité. Il est ainsi relativement courant de prendre $h_{uv} = 1/\sqrt{w_{uv}}$ [Bougleux *et al.*, 2007, Elmoataz *et al.*, 2008, Ta, 2009]. L'opérateur de différence pondérée s'exprime alors par la formule suivante [8] :

$$(d_w f)(u, v) = \sqrt{w_{uv}}(f(v) - f(u)) \quad . \tag{1.17}$$

L'opérateur de différence pondérée est lié à son adjoint par la relation suivante :

$$\langle d_w f, F \rangle = \langle f, d_w^* F \rangle \quad , \tag{1.18}$$

avec $\langle \cdot, \cdot \rangle$ l'opérateur produit scalaire (voir equations (1.1) et (1.2)).

L'opérateur adjoint de l'opérateur de différence pondérée $d_w^* : \mathcal{H}(E) \rightarrow \mathcal{H}(V)$ est ainsi défini pour chaque nœud u par :

$$(d_w^* F)(u) = \sum_{v \sim u} \sqrt{w_{uv}}(F(v, u) - F(u, v)) \quad . \tag{1.19}$$

Il est alors possible de définir l'opérateur de divergence $-d_w^*$, lequel mesure le flot sortant en chaque nœud $u \in V$ d'une fonction $F \in \mathcal{H}(E)$.

1.7.1.2 Opérateur gradient pondéré

L'opérateur gradient pondéré $\nabla_w f : \mathcal{H}(V) \rightarrow \mathbb{R}^m$ permet d'évaluer la régularité d'une fonction $f \in \mathcal{H}(V)$. Il est défini en chaque nœud $u \in V$ comme étant le vecteur des différences pondérées pour l'ensemble des arêtes incidentes à u :

$$(\nabla_w f)(u) = ((d_w f)(u, v))_{v \sim u}^T \quad . \tag{1.20}$$

Par la suite, c'est essentiellement la norme L_q de ce gradient qui sera utilisée :

$$\|(\nabla_w f)(u)\|_q = \left[\sum_{v \sim u} \left(\sqrt{w_{uv}} |f(v) - f(u)| \right)^q \right]^{1/q} \quad . \tag{1.21}$$

8. La notation de l'équation (1.16) est utilisée par analogie avec la définition de l'opérateur traditionnel de dérivée, dont l'opérateur de différence pondérée est le pendant lorsque appliqué à des graphes.

1.7.1.3 Opérateur p-Laplacien pondéré isotrope et anisotrope

L'opérateur p-Laplacien pondéré est un opérateur de second ordre dont le rôle est de mesurer la régularité des variations d'une fonction $f \in \mathcal{H}(V)$. Il est défini pour tout réel p tel que $0 < p < +\infty$, paramètre dont le rôle est de pénaliser les fortes variations de f.

Dans sa version isotrope, le p-Laplacien pondéré $\Delta^i_{w,p} : \mathcal{H}(V) \to \mathcal{H}(V)$ est défini par :

$$\Delta^i_{w,p} f = d^*_w \left(\|\nabla_w f\|_2^{p-2} d_w f \right) \quad . \tag{1.22}$$

Le p-Laplacien pondéré anisotrope $\Delta^a_{w,p} : \mathcal{H}(V) \to \mathcal{H}(V)$ est quant à lui défini par :

$$\Delta^a_{w,p} f = d^*_w \left(|d_w f|^{p-2} d_w f \right) \quad . \tag{1.23}$$

En un nœud $u \in V$, ces opérateurs s'expriment par :

$$\begin{aligned}
(\Delta^i_{w,p} f)(u) &= \sum_{v \sim u} \sqrt{w_{uv}} \left(\frac{(d_w f)(vu)}{\|(\nabla_w f)(v)\|_2^{2-p}} - \frac{(d_w f)(uv)}{\|(\nabla_w f)(u)\|_2^{2-p}} \right) \quad , \\
&= \sum_{v \sim u} w_{uv} \left(\frac{f(u) - f(v)}{\|(\nabla_w f)(v)\|_2^{2-p}} - \frac{f(v) - f(u)}{\|(\nabla_w f)(u)\|_2^{2-p}} \right) \quad , \\
&= \sum_{v \sim u} w_{uv} \left(\|(\nabla_w f)(v)\|_2^{p-2} + \|(\nabla_w f)(u)\|_2^{p-2} \right) (f(u) - f(v)) \quad ,
\end{aligned} \tag{1.24}$$

et :

$$\begin{aligned}
(\Delta^a_{w,p} f)(u) &= \sum_{v \sim u} \sqrt{w_{uv}} \left(\frac{(d_w f)(vu)}{|(d_w f)(vu)|^{2-p}} - \frac{(d_w f)(uv)}{|(d_w f)(uv)|^{2-p}} \right) \quad , \\
&= \sum_{v \sim u} w_{uv}^{p/2} \left(\frac{f(u) - f(v)}{|f(u) - f(v)|^{2-p}} - \frac{f(v) - f(u)}{|f(v) - f(u)|^{2-p}} \right) \quad , \\
&= 2 \sum_{v \sim u} w_{uv}^{p/2} |f(u) - f(v)|^{p-2} (f(u) - f(v)) \quad .
\end{aligned} \tag{1.25}$$

1.7.2 Fonctionnelles de régularisation

Comme évoqué plus tôt, le rôle d'une fonctionnelle de régularisation est de mesurer la régularité d'une fonction. Dans ses travaux, Ta reprend la famille de fonctionnelles isotropes proposée dans [Bougleux et Elmoataz, 2005, Elmoataz *et al.*, 2008]. Celles-ci sont définies sur les graphes et utilisent la norme L_2 du gradient pondéré (1.21) afin de proposer une mesure inspirée de la variation totale [9].

La variation totale correspond à l'amplitude cumulée de l'ensemble des variations d'une fonction à valeur réelles $g : \mathbb{R}^m \to \mathbb{R}$. Elle est définie sur un intervalle borné $\Omega \in \mathbb{R}^m$ par :

$$TV_\Omega(g) = \int_\Omega |g\prime(x)| dx \quad . \tag{1.26}$$

9. Le principe de variation totale sera par la suite abrégé par TV, de l'anglais *Total Variation*.

Elle peut être adaptée aux graphes de similarité en considérant la norme du gradient en chaque nœud. Soit $G = (V, E, w)$ un graphe pondéré et $f \in \mathcal{H}(V)$ une fonction à valeur réelles définie sur l'ensemble des nœuds. Cette famille de fonctionnelles s'exprime alors pour tout $0 < p < +\infty$ par :

$$\begin{aligned} \mathcal{R}^i_{w,p} &= \sum_{u \in V} \|(\nabla_w f)(u)\|_2^p \quad , \\ &= \sum_{u \in V} \left[\sum_{v \sim u} w_{uv} \left(f(v) - f(u) \right)^2 \right]^{p/2} \quad . \end{aligned} \tag{1.27}$$

[Ta, 2009] propose une nouvelle famille de fonctionnelles anisotropes de régularisation, basée sur la norme L_q du gradient pondéré. Elle s'exprime pour $0 < p < +\infty$ et $q = p$ par :

$$\begin{aligned} \mathcal{R}^a_{w,p} &= \sum_{u \in V} \|(\nabla_w f)(u)\|_p^p \quad , \\ &= \sum_{u \in V} \sum_{v \sim u} w_{uv}^{p/2} \left| f(v) - f(u) \right|^p \quad . \end{aligned} \tag{1.28}$$

Ces deux familles de fonctionnelles peuvent s'exprimer de manière générale par :

$$\mathcal{R}^*_{w,q,\phi} = \sum_{u \in V} \phi(\|(\nabla_w f)(u)\|_q) \quad , \tag{1.29}$$

avec $\phi(\cdot)$ le noyau de régularisation permettant de pénaliser les fortes variations de la fonction f au voisinage d'un nœud.

1.7.3 Première classe de modèles variationnels

À partir des éléments présentés précédemment, une première classe de modèles variationnels peut être définie [Ta *et al.*, 2007a, Ta *et al.*, 2007b]. Le critère de régularisation utilisé est inspiré de la variation totale. Cette méthode est notamment appliquée à plusieurs problèmes d'interpolation. Pour rappel, en reconnaissance de formes, l'interpolation consiste à attribuer à chacune des données une valeur à partir d'un ensemble de marqueurs initiaux. Un exemple typique de problème d'interpolation – que cet algorithme est capable de traiter – est la colorisation d'images, où le but est d'attribuer un code couleur à chacun des pixels à partir de marqueurs colorimétriques [Lezoray *et al.*, 2008, Ta, 2009].

1.7.3.1 Formulation

On considère un graphe de similarité $G = (V, E, w)$ associé à l'image à traiter, et $f_0 \in \mathcal{H}(V)$ la fonction initiale représentative des marqueurs initiaux.

Cette première classe de modèles s'exprime par :

$$\min_{f : V \to \mathbb{R}} \left\{ \mathcal{E}_1(f, f_0, \lambda) = \mathcal{R}^*_{w,q,\phi} + \sum_{u \in V} \frac{\lambda}{2} \|f(u) - f_0(u)\|_2^2 \right\} \quad , \tag{1.30}$$

avec $f \in \mathcal{H}(V)$ la solution, résultat de l'interpolation.

Suivant les valeurs prises par le paramètre q de la fonctionnelle de régularisation ainsi que l'expression du noyau de régularisation $\phi(\cdot)$ utilisé, deux modèles de régularisation peuvent être générés à partir de cette classe. En particulier, en utilisant $\phi(s) = \frac{1}{p}s^p$ et $q = 2$, le modèle de régularisation obtenu est appelé p-TV isotrope, tandis qu'avec $\phi(s) = \frac{1}{2p}s^p$ et $p = q$, le modèle est p-TV anisotrope.

Le problème de minimisation (1.30) associé à cette classe de modèles est convexe si $p \geq 1$. Il admet donc un minimum global. La dérivée partielle de l'énergie $\mathcal{E}_1(f, f_0, \lambda)$ par rapport à f est alors nulle en tout sommet $u \in V$:

$$\frac{\partial \mathcal{E}_1(f, f^0, \lambda)}{\partial f(u)} = 0 \quad , \tag{1.31}$$

équation que l'on peut réécrire :

$$\frac{\partial \mathcal{E}_1(f, f^0, \lambda)}{\partial f(u)} = \sum_{v \sim u} \alpha_{uv}^{q,\phi,f}(f(u) - f(v)) \cdot |f(u) - f(v)|^{q-2} + \lambda(f(u) - f^0(u)) = 0 \quad , \tag{1.32}$$

avec :

$$\alpha_{uv}^{q,\phi,f} = w_{uv}^{q/2} \left(\frac{\phi'(\|(\nabla_w f)(u)\|_q)}{\|(\nabla_w f)(u)\|_q^{q-1}} + \frac{\phi'(\|(\nabla_w f)(v)\|_q)}{\|(\nabla_w f)(v)\|_q^{q-1}} \right) \quad . \tag{1.33}$$

Pour des raisons de stabilité numérique, il convient d'utiliser des versions régularisées de la norme du gradient et de la valeur absolue. Les opérateurs utilisés par la suite sont donc :

$$\|(\nabla_w f)(u)\|_q = \sqrt{\|(\nabla_w f)(u)\|_q + \varepsilon} \quad , \tag{1.34}$$

$$|f(u) - f(v)| = |f(u) - f(v)| + \varepsilon \quad , \tag{1.35}$$

avec ε une valeur positive proche de zéro, mais la notation initiale est conservée par mesure de clarté.

La solution du problème de minimisation peut être approximée par un algorithme itératif [10]. En réécrivant l'équation (1.32) sous la forme suivante :

$$f(u) = \frac{\lambda f_0(u) + \sum_{v \sim u} \alpha_{uv}^{q,\phi,f} |f(u) - f(v)|^{q-2} f(v)}{\lambda + \sum_{v \sim u} \alpha_{uv}^{q,\phi,f} |f(u) - f(v)|^{q-2}} \quad \forall u \in V \quad , \tag{1.36}$$

il est possible d'appliquer l'algorithme itératif de Gauss-Jacobi :

$$\begin{cases} f^0(u) = f_0(u) \\ f^{n+1}(u) = \dfrac{\lambda f_0(u) + \sum_{v \sim u} \alpha_{uv}^{q,\phi,f^n} |f^n(u) - f^n(v)|^{q-2} f(v)}{\lambda + \sum_{v \sim u} \alpha_{uv}^{q,\phi,f^n} |f^n(u) - f^n(v)|^{q-2}} \end{cases} \quad \forall u \in V. \tag{1.37}$$

La condition d'arrêt de l'algorithme peut être un nombre prédéfini d'itérations, ou encore l'atteinte d'un certain niveau de convergence, avec $\|f^{n+1} - f^n\| < \tau$.

10. D'autres méthodes sont également applicables. Nous pouvons notamment citer les approches à base de coupes de graphes [Kolmogorov et Zabih, 2004] ou encore celles basées sur une expression duale du problème de minimisation [Hidane *et al.*, 2012, Couprie *et al.*, 2013].

1.7.3.2 Application à la segmentation semi-supervisée

Ta propose de traiter des problèmes de classification et de segmentation de données semi-supervisées à partir de cet algorithme d'interpolation. L'objectif est alors d'attribuer à chacune des données non plus une valeur numérique mais une classe (ou un label) à partir de marqueurs positionnés sur un ensemble d'éléments représentatifs de chacune des classes à identifier.

On suppose que l'image est constituée de c régions à identifier, nombre connu *a priori*. Soit C_i, $i = 1 \dots c$, l'ensemble des nœuds initialement marqués comme appartenant à la classe i, et $V_0 = \cup C_i$ l'ensemble des nœuds disposant d'un marqueur initial. La fonction initiale $f_0 \in \mathcal{H}^c(V)$ associe à chaque nœud un vecteur composé de c valeurs réelles : $f_0 = (f_{0,i})_{i=1\dots c}$, avec :

$$f_{0,i}(u) = \begin{cases} 1 & \text{si } u \in C_i \\ 0 & \text{sinon} \end{cases} . \tag{1.38}$$

Afin de permettre la reconnaissance de c classes, c processus de régularisation de graphe basés sur l'équation (1.37) sont exécutés en parallèle, chacun cherchant à calculer une fonction $f_i : V \to [0, 1]$ solution du problème de minimisation (1.30) étant donné $f_{0,i}$.

Nous obtenons ainsi pour le modèle p-TV isotrope :

$$\begin{cases} f_i^0(u) = f_{0,i}(u) \\ f_i^{n+1}(u) = \dfrac{\lambda f_{i,0}(u) + \sum_{v\sim u} w_{uv} \left(\|(\nabla_w f_i^n)(u)\|_2^{p-2} + \|(\nabla_w f_i^n)(v)\|_2^{p-2} \right) f_i^n(v)}{\lambda + \sum_{v\sim u} w_{uv} \left(\|(\nabla_w f_i^n)(u)\|_2^{p-2} + \|(\nabla_w f_i^n)(v)\|_2^{p-2} \right)} \quad \forall u \in V, \end{cases} \tag{1.39}$$

et :

$$\begin{cases} f_i^0(u) = f_{0,i}(u) \\ f_i^{n+1}(u) = \dfrac{\lambda f_{i,0}(u) + \sum_{v\sim u} w_{uv}^{p/2} |f_i^n(u) - f_i^n(v)|^{p-2} f_i^n(v)}{\lambda + \sum_{v\sim u} w_{uv}^{p/2} |f_i^n(u) - f_i^n(v)|^{p-2}} \quad \forall u \in V, \end{cases} \tag{1.40}$$

pour le modèle p-TV anisotrope.

Enfin, le résultat final de la segmentation est calculé par vote majoritaire, les valeurs représentées par les différents f_i pouvant être interprétées comme un potentiel d'appartenance de chaque nœud à la classe i.

1.7.4 Seconde classe de modèles variationnels

Dans [Ta, 2009], l'auteur propose une seconde classe de modèles variationnels. Elle se distingue de la précédente de deux façons. D'une part, il s'agit d'une classe dédiée à la segmentation de données. La solution ainsi recherchée est une fonction indicatrice de la segmentation qui est donc binaire. D'autre part, le critère d'attache aux données n'est plus limité à la notion de variation totale. Ta propose ainsi d'utiliser deux implémentations différentes de cette classe de modèles variationnels, basées respectivement sur les modèles

de Rudin, Osher et Fatemi [11] [Rudin *et al.*, 1992] et de Chan et Vese [12] [Chan et Vese, 2001], implémentations que nous détaillerons par la suite. La forme générale de cette classe de problèmes s'exprime par :

$$\min_{f:V\to\{0,1\}} \left\{ \mathcal{E}_2(f, f_0, \lambda) = \mathcal{R}^*_{w,q,\phi} + \sum_{u\in V} \lambda g(f_0)(u) f(u) \right\} \quad , \tag{1.41}$$

avec $g(\cdot)$ la fonction jouant le rôle de terme d'attache aux données.

Afin de pouvoir aboutir à un algorithme itératif similaire à celui utilisé pour résoudre la première classe de problème, Ta propose une version relaxée du problème de minimisation (1.41), dans laquelle les solutions ne sont plus binaires, mais définies dans un domaine continu, à savoir $[0, 1]$:

$$\min_{f:V\to[0,1]} \left\{ \tilde{\mathcal{E}}_2(f, f_0, \lambda) = \mathcal{R}^*_{w,q,\phi} + \sum_{u\in V} \lambda g(f_0)(u) f(u) \right\} \quad . \tag{1.42}$$

Ta démontre, à l'aide de la formule de la co-aire [Chambolle, 2005] que, dans le cas où $\phi(s) = \frac{1}{p}s^p$ et $p = q = 1$, le problème de minimisation (1.42) est convexe et que la solution du problème relaxé est aussi solution du problème initial (1.41). Il est ainsi possible d'approximer une solution du problème de segmentation à l'aide du même algorithme itératif utilisé pour la première classe, basé sur l'annulation du gradient :

$$\begin{cases} f^0(u) = m(u) \\ f^{n+1}(u) = \dfrac{\sum_{v\sim u} \sqrt{w_{uv}} |f^n(u) - f^n(v)|^{-1} f^n(v) - \lambda g(f_0)(u)}{\sum_{v\sim u} \sqrt{w_{uv}} |f^n(u) - f^n(v)|^{-1}} \end{cases} \quad \forall u \in V \quad , \tag{1.43}$$

avec $m : V \to \{0, 1\}$ le marqueur initial, une segmentation grossière que l'algorithme a pour rôle d'affiner au fil des itérations.

Du fait de la relaxation du problème, il est cependant nécessaire de projeter la solution calculée après chaque itération dans l'intervalle $[0, 1]$ tel que :

$$f^n(u) = \begin{cases} 0 & \text{si } f^n(u) < 0 \\ 1 & \text{si } f^n(u) > 1 \\ f^n(u) & \text{sinon} \end{cases} \quad . \tag{1.44}$$

1.7.4.1 Modèle de Rudin, Osher et Fatemi

Le modèle de Rudin, Osher et Fatemi s'inspire d'une méthode de débruitage d'images qui se base sur une mesure de la variation totale.

Cette mesure part du principe qu'un pixel de bruit présente un niveau de gris très différent de ses voisins. Elle est ainsi définie comme la somme des gradients d'intensité en chaque point de l'image. Sa minimisation entraine une régularisation de la fonction qui associe à chaque pixel son niveau de gris, et donc un débruitage de l'image.

11. Ce modèle sera pas la suite dénommé « modèle de ROF ».
12. Ce modèle sera pas la suite dénommé « modèle de CV ».

Initialement proposée dans [Rudin *et al.*, 1992], cette méthode s'exprime dans le domaine continu par :

$$\mathcal{J}_{ROF}(f, f_0, \lambda) = \int_\Omega \|\nabla f\|_1 + \lambda \int_\Omega (f(x) - f_0(x))^2 \, dx \quad , \tag{1.45}$$

avec $f_0 : \Omega \subset \mathbb{R}^m \rightarrow [0, 1]$ la fonction initiale représentant l'image, et f la solution recherchée.

Afin de transformer cette fonctionnelle d'énergie en un problème de minimisation appartenant à la seconde classe de modèles variationnels, il est nécessaire de procéder à une reformulation du terme d'attache aux données. En se restreignant à une fonction f binaire, celui-ci peut se réécrire :

$$(f - f_0)^2 = f^2 - f f_0 + f_0^2 = f - f f_0 + f_0^2 = (1 - f_0) f + f_0^2 \quad , \tag{1.46}$$

ce qui permet de reformuler la fonctionnelle de la façon suivante :

$$\mathcal{J}_{ROF}(f, f_0, \lambda) = \int_\Omega \|\nabla f\|_1 + \lambda \int_\Omega \Big((1 - f_0(x)) f(x) + f_0(x)^2 \Big) \quad . \tag{1.47}$$

Étant donné que le terme $f_0(x)^2$ est constant, la solution f minimisant la fonctionnelle d'énergie 1.47 est aussi la solution du problème de minimisation de la fonctionnelle suivante :

$$\mathcal{J}_{ROF}(f, f_0, \lambda) = \int_\Omega \|\nabla f\|_1 + \lambda \int_\Omega (1 - f_0(x)) f(x) dx \quad . \tag{1.48}$$

Ce modèle étant, de par sa formulation, un cas particulier du problème (1.41), Ta a proposé d'étendre la formulation de ce modèle aux graphes :

$$\min_{f:V \rightarrow \{0,1\}} \left\{ \mathcal{E}_{ROF}(f, f_0, \lambda) = \mathcal{R}_{w,1} + \sum_{u \in V} \lambda g(f_0)(u) f(u) \right\} \quad , \tag{1.49}$$

avec :

$$g(f_0) = (1 - 2 f_0) \quad . \tag{1.50}$$

1.7.4.2 Modèle de Chan et Vese

Selon une logique similaire, Ta étend aux graphes la méthode de segmentation par contours actifs proposée dans [Chan et Vese, 2001].

L'objectif de l'approche proposée par Chan et Vese est de partitionner une image en deux régions composées de pixels dont les niveaux de gris respectifs sont homogènes. Cette méthode nécessite la définition d'un contour initial par l'utilisateur afin de permettre à l'algorithme d'obtenir une première évaluation des niveaux de gris moyens des deux régions. Le résultat de la segmentation est ensuite représenté sous la forme d'un ensemble de niveaux que l'algorithme déforme itérativement selon des critères d'homogénéité. Après chaque itération, les niveaux de gris moyens sont ré-évalués afin de correspondre au résultat qui vient d'être calculé, permettant ainsi une meilleure modélisation du problème. Étant donné

que cette méthode se base sur une caractéristique basée région, elle permet la détection d'objets ne présentant pas de contours nets.

Comme la méthode de ROF, l'approche proposée par Chan et Vese repose sur un problème de régularisation. Soit $f_0 : \Omega \subset \mathbb{R}^m \to [0,1]$ la fonction initiale représentant l'image. Le modèle de Chan et Vese s'exprime dans le domaine continu à l'aide de la fonctionnelle d'énergie suivante :

$$\begin{aligned} \mathcal{J}_{CV}(f, f_0, \lambda_1, \lambda_2) = & \int_\Omega \|\nabla f\|_1 \\ & + \lambda_1 \int_\Omega f(x)(c_1 - f_0(x))^2 dx \\ & + \lambda_2 \int_\Omega (1 - f(x))(c_2 - f_0(x))^2 dx \quad , \end{aligned} \tag{1.51}$$

avec $f : \Omega \subset \mathbb{R}^m \to \{0,1\}$ la solution recherchée indicatrice de la segmentation, et $c_1, c_2 \in \mathbb{R}$ deux variables représentant la valeur moyenne de la fonction f_0 respectivement à l'intérieur et à l'extérieur de la segmentation.

Les deux paramètres λ_1 et λ_2 permettent de pondérer l'influence du terme d'attache aux données vis-à-vis du terme de régularisation ainsi que l'importance relative de l'attache aux données pour chacune des deux régions. Cela permet la définition de contraintes concernant l'étendue des niveaux de gris autorisés dans chaque région. Étant donné que, dans l'essentiel de leurs tests, Chan et Vese ont fixés $\lambda_1 = \lambda_2$, Ta a choisi de ne conserver qu'un unique paramètre λ. La fonctionnelle d'énergie peut ainsi être reformulée de la façon suivante :

$$\begin{aligned} \mathcal{J}_{CV}(f, f_0, \lambda) = & \int_\Omega \|\nabla f\|_1 \\ & + \lambda \int_\Omega f(x)(c_1 - f_0(x))^2 + (1 - f(x))(c_2 - f_0(x))^2 dx \quad . \end{aligned} \tag{1.52}$$

Afin d'illustrer le fonctionnement du terme d'attache aux données, considérons que l'image à segmenter est composée de deux régions homogènes de niveaux de gris respectifs c_1 et c_2. Ces deux régions seront respectivement identifiées par les valeurs 1 et 0 de la fonction indicatrice f. Considérons un pixel x appartenant à la première région pour lequel nous avons :

$$(f_0(x) - c_1)^2 \approx 0 \quad \text{et} \quad (f_0(x) - c_2)^2 \gg 0 \quad . \tag{1.53}$$

Si x est correctement identifié par l'algorithme de segmentation comme appartenant à la première région, $f(x) = 1$. Dans ces conditions le terme d'attache aux données contribue faiblement à la fonctionnelle d'énergie (1.52) :

$$f(x)(c_1 - f_0(x))^2 + (1 - f(x))(c_2 - f_0(x))^2 \approx 0 \quad . \tag{1.54}$$

En revanche s'il est identifié comme appartenant à la seconde région, $f(x) = 0$. Le terme d'attache aux données contribue donc fortement à la valeur de la fonctionnelle d'énergie (1.52) :

$$f(x)(c_1 - f_0(x))^2 + (1 - f(x))(c_2 - f_0(x))^2 \gg 0 \quad . \tag{1.55}$$

Un raisonnement similaire peut être appliqué aux pixels de la seconde région, illustrant ainsi le fait que tout pixel mal classé accroit l'énergie associée à la solution.

Afin de transformer cette fonctionnelle d'énergie en un problème de minimisation représenté à l'aide de la seconde classe de modèles variationnels, il est de nouveau nécessaire de procéder à une reformulation du terme d'attache aux données :

$$f \cdot (c_1 - f_0)^2 + (1 - f) \cdot (c_2 - f_0)^2 = \Big((c_1 - f_0)^2 - (c_2 - f_0)^2\Big) f + (c_2 - f_0)^2 \quad . \quad (1.56)$$

Les variables c_1 et c_2 seront par la suite calculées avant chaque itération de l'algorithme et sont donc considérées comme constantes lors des calculs liés à l'annulation du gradient. Le dernier terme de l'équation (1.56) n'influant pas sur le calcul du gradient, il peut donc être ignoré. Il est ainsi possible de reformuler la fonctionnelle (1.52) par :

$$\mathcal{J}_{CV}(f, f_0, \lambda) = \int_\Omega \|\nabla f\|_1 + \lambda \int_\Omega f(x) \cdot g(f_0)(x) dx \quad , \quad (1.57)$$

avec :

$$g(f_0) = (c_1 - f_0)^2 - (c_2 - f_0)^2 \quad . \quad (1.58)$$

Cette fonctionnelle étant de nouveau un cas particulier du problème (1.41), elle peut être généralisée en un problème de minimisation défini sur les graphes :

$$\min_{f:V\to\{0,1\}} \left\{ \mathcal{E}_{CV}(f, f_0, \lambda) = \mathcal{R}_{w,1} + \sum_{u\in V} \lambda g(f_0)(u) f(u) \right\} \quad . \quad (1.59)$$

1.8 Conclusion

Comme nous avons pu le voir, il existe de nombreuses façons de représenter une image à l'aide d'un graphe. Cependant, si l'influence de certaines topologies sur la qualité des résultats a été constatée à quelques reprises, aucune théorie à ce sujet n'a été formulée.

Les trois premières familles de méthodes de segmentation d'images à base de graphes évoquées (arbres de couverture minimale, coupe minimales et marche aléatoire) représentent des sujets de recherche tout à fait dignes d'intérêt. En effet, elles dépendent exclusivement de la topologie des graphes et reposent sur d'importants résultats issue de la théorie des graphes dont le comportement, lorsqu'appliqués à des problèmes de vision par ordinateur, est encore mal maîtrisé. Ces méthodes sont cependant relativement lourdes. Malgré certaines optimisations et relaxations, les algorithmes de la théorie des graphes qu'elles emploient ne peuvent pas s'exécuter en temps linéaire (par rapport au nombre de nœuds du graphe). Ces méthodes sont donc, encore aujourd'hui, difficilement applicables à de réels problèmes de vision par ordinateur.

Dans cette thèse, nous nous intéresserons aux méthodes de segmentation par régularisation de graphes proposées par Ta [Ta, 2009]. Celles-ci peuvent s'exécuter en temps linéaire, mais s'inspirent surtout de théories beaucoup plus récentes, pour lesquelles de nombreuses options restent encore à explorer. Afin d'en améliorer la précision, nous proposons d'améliorer une de ces méthodes afin de lui permettre de tirer parti de caractéristiques de textures.