Traitement de l'image et Géométrie Discrète

Rapport de projet

Arbre des coupes

Groupe :
Sébastien Aubry
Kelana Saint-Marc

Encadrant :
Benoît Naegel

22 janvier 2024

Table des matières

1 Introduction

Notre sujet s'intéresse à l'utilisation d'un arbre de partition binaire pour la représentation d'images. L'intérêt étant de pouvoir segmenter ou filtrer l'image selon certaines régions. Pour analyser ce sujet nous avons étudié l'article de P. Salembier [1].
Cet article présente le concept d'arbre de partition binaire ainsi qu'une méthode de construction. Il expose également différents cas d'usage de l'arbre, à savoir la recherche d'information, la segmentation et le filtrage. Notre travail consiste en l'implémentation et l'expérimentation autour des méthodes présentées dans l'article. Tout ceci en prenant soin de renseigner nos résultats pour en discuter.
Ce rapport consiste donc en une présentation des différents sujets et méthodes abordés dans l'article, suivi d'une description de notre mise en oeuvre, terminé par un exposé de nos résultats.

2 Méthodologie de l'article

2.1 Idée de base

L'objectif était de combiner les deux approches que constituent la segmentation et les opérateurs connexes basés sur Max-Tree. En effet, la segmentation permet d'être assez flexibles sur la définition des régions tandis que l'approche Max-Tree des opérateurs connexes permet la sélection des régions souhaitées. C'est pourquoi ils présentent la construction de l'arbre binaire en utilisant la segmentation ainsi que son traitement avec la méthode utilisée pour les opérateurs connexes.

2.2 Méthode

2.2.1 Construction de l'arbre

Pour la construction de l'arbre, ils utilisent un RAG (Region Adjacency Graph) comme représentation de l'image. Cela permet de garder la trace des différentes étapes de rassemblement des régions.
Ils commencent par partitionner l'image avec des zones plates avant de rassembler les différentes zones. Ensuite ils construisent le RAG pour prendre en compte le nombre et le voisinage des régions.
Une fois le RAG obtenu ils rassemblent les zones deux à deux en fonction de leur ressemblance. Ils continuent à rassembler les régions jusqu'à ce qu'il n'en reste plus qu'une dans le RAG. Le critère de ressemblance présenté dans l'article est l'homogénéité de la couleur (YUV).
Une fois toutes les régions traitées l'arbre est complet.

2.2.2 Traitement de l'arbre

Segmentation L'article présente deux approches, l'approche directe et celle par marqueurs. L'approche directe consiste à affecter rassembler les noeuds de l'arbre jusqu'à atteindre un critère. Concernant le critères ils citent le nombre de régions et le PSNR entre l'image originale et l'image créée. La construction de l'arbre a fixé l'ordre de fusion

des zones donc l'ordre n'est pas modifié par le critère de fusion (ici le critère de terminaison). Étant donné que l'arbre a été construit selon le même critère que celui utilisé pour la segmentation, il suffit de désactiver progressivement les noeuds en suivant la séquence de fusion jusqu'à rencontrer le critère de terminaison.
Outre cette méthode, l'article présente aussi l'approche morphologique ou "Marqueur et propagation" qui se présente comme l'application de marqueurs puis la définition de la zone d'influence de ce marqueur.

Filtrage Ils présentent la technique d'élagage de l'arbre. L'objectif de cette méthode est de ne représenter que certains noeuds. Pour cela on va parcourir l'arbre de bas en haut jusqu'à arriver à la racine. Tout au long de ce parcours on va prendre des décisions quand la conservation ou l'élimination du noeud ou de la feuille. Une fois toutes les décisions prises, l'arbre nous permet de représenter l'image sans les parties indésirables.

3 Travail effectué

3.1 Structures de données

Graphe d'adjacence Nous avons structuré notre RAG de façon à représenter les noeuds et les arêtes. Nous stockons les noeuds représentant les régions dans une liste. Puis nous stockons les arêtes dans une autre liste. Ces arêtes sont une paire d'index de noeuds (index de la position du noeud dans la liste de noeud).

Arbre des coupes Pour structurer notre arbre nous avons mis en place une classe qui implémente différentes méthodes. Cette classe permet d'initialiser un noeud et d'y stocker une liste de pixels, une taille de région, un accès à 2 autres noeuds enfants, ainsi qu'une option pour marquer ces 2 noeuds enfants. Il y a une méthode qui permet d'ajouter les 2 noeuds enfants liés au noeud principal. Une autre méthode qui permet de retourner la liste de tous les pixels accessibles en parcourant les noeuds rattachés au noeud principal. Une méthode pour marquer un noeud enfant. Une méthode pour enlever la marque sur tous les noeuds accessibles depuis le noeud principal. Et enfin, une méthode permettant de retourner une image traitée avec l'arbre en fonction d'une image et des branches accessibles depuis le noeud racine de l'arbre.

3.2 Implémentation

3.2.1 Segmentation de l'image

On cherche à partitionner l'image en zones plates. Pour segmenter l'image nous nous servons de l'algorithme de k-means afin de définir les différentes régions en fonction du nombre de zones souhaitées.
Une fois l'algorithme k-means appliqué on crée un nouvelle image avec pour valeur de pixels le label attribué par k-means. Ainsi nous avons une image composée de plusieurs zones de pixels définies par leur label de couleur.
Une fois cette image segmentée obtenue nous pouvons l'utiliser pour construire le graphe d'adjacence.

3.2.2 Construction du Graphe d'adjacence

On commence par créer un noeud par région de l'image. Une région étant un ensemble de pixels connexes de même label.
Pour créer les noeuds de l'arbre on utilise un système de marquage en passant par une image auxiliaire. Pour chaque pixel non marqué on crée une nouvelle région que l'on propage à tous les pixels connexes de même couleur. Et ainsi de suite jusqu'à avoir marqué tous les pixels de l'image. A chaque fois qu'un pixel est marqué il est également ajouté au noeud correspondant à son marquage. Une fois tous les noeuds créés on passe à la création des arêtes les reliant.
Pour ce faire, on parcourt tous les pixels de l'image créée à la création des noeuds. A chaque voisinage de région différente on ajoute un nouvel arc (si l'arc n'existe pas encore). Une fois tous les arcs obtenus, le graphe est complet et peut être utilisé pour la construction de l'arbre.

3.2.3 Construction de l'arbre

Une fois le RAG complet on commence la création de l'arbre. Elle se centre autour des arêtes du RAG. En effet, on va chercher à dé-construire le RAG en fonction de l'ordre des arêtes pour construire notre arbre, petit à petit.
On commence par trouver l'arête avec le critère de fusion le plus élevé. Le critère de fusion est obtenu à partir d'une fonction. Une fois l'arête sélectionnée on crée un noeud dans l'arbre avec pour feuilles les deux extrémités de l'arête, avant de la supprimer du RAG.
Une fois ce noeud créé on l'ajoute également au RAG et on remplace les extrémités de l'arête par le nouveau noeud dans toutes les arêtes encore présentes dans le RAG. On répète cette opération jusqu'à ce que le RAG n'ait plus aucune arête.

3.2.4 Traitement de l'arbre

Pour le traitement de l'arbre, nous avons mis en place 2 fonctions filtre d'exemple. Pour traiter un arbre, il suffit de bloquer/marquer ses branches pour que celles-ci ne soient pas accessible lors du calcul de l'image avec la méthode get_image(image).

Premier filtre Nous avons mis en place un filtre afin de traiter l'arbre en fonction d'une certaine valeur de gris. Si la valeur de gris d'une région est inférieure à la valeur de gris cible, alors on bloque la branche liée à cette région.

Deuxième filtre Pour ce deuxième filtre nous avons cherché à faire ressortir les régions de taille supérieure à un certain seuil. Pour cela nous regardons le nombre de pixels dans chaque noeud afin d'élaguer l'arbre. Si le noeud contient suffisamment de pixels il sera conservé, sinon sa branche sera bloquée.

4 Résultats obtenus

FIGURE 1 – Image de base utilisé pour les tests de notre programme

FIGURE 2 – Image de base après une segmentation en différente région à partir de zone plate

FIGURE 3 – Image obtenue après création de l'arbre en choisissant d'afficher uniquement la branche de droite

FIGURE 4 – Image obtenue en appliquant un filtre sur l'arbre. Filtre qui affiche uniquement les régions dont le niveau de gris est supérieur à 64

FIGURE 5 – Image obtenue en appliquant un filtre sur l'arbre. Filtre qui affiche uniquement les régions d'une taille supérieur à 10 000 pixels

5 Conclusion

Notre implémentation n'est pas tout à fait fidèle à celle de l'article mais nous avons quand même pu étudier les différents concepts qui y étaient évoqués. Nous avons notamment pu coder un partitionnement et deux méthodes de filtrage de l'image. Notre code a un fort potentiel d'amélioration, tant dans nos méthodes de traitement que dans nos méthodes de création. En effet, ces deux parties étant très liées, il aurait été intéressant de tester la création d'arbres suivant des critères différents, et les traitements adéquats.

Références

[1] Philippe Salembier et Luis Garrido. « Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval ». In : *Image Processing, IEEE Transactions on* 9 (mai 2000), p. 561-576. DOI : 10.1109/83.841934.