Université Paris Cité

Rapport de TP
Opérateurs Connexes et Segmentation

Introduction à l'Analyse d'Image - M1

Université Paris Cité

Année universitaire : 2024-2025

Étudiante :
Anaïs Kadic

Enseignant :
M. Camille Kurtz

Table des matières

1 Introduction 2

2 Théorie des Opérateurs Connexes et Segmentation 3

2.1 Adjacence et Relations de Voisinage . . . 3
2.2 Adjacence et Relations de Voisinage . . . 3

3 Résultats de l'Étiquetage des Composantes Connexes 4

3.1 Principe de l'Algorithme . . . 4
3.2 Exemple d'Application . . . 4

4 Optimisation de l'Étiquetage des Composantes Connexes : Algorithme en Deux Passes 5

4.1 Implémentation de l'Algorithme en Deux Passes . . . 5
4.2 Formulation Mathématique . . . 6
4.3 Comparaison entre DFS et l'Algorithme en Deux Passes . . . 6
4.4 Optimisation : Gestion des Trous et Structure Hiérarchique . . . 6

5 Filtrage d'Aire des Composantes Connexes 7

5.1 Objectif . . . 7
5.2 Définition Mathématique du Filtrage d'Aire . . . 7
5.3 Exemple d'Application . . . 7
5.4 Choix du Type de Label . . . 8
5.5 Conclusion . . . 8

6 Conclusion 9

1 Introduction

Lorsque l'on parle d'analyse d'images, on évoque les techniques permettant d'extraire, traiter et interpréter l'information visuelle contenue dans une image. Cela repose notamment sur la détection et la segmentation des structures présentes dans une image binaire. En effet, nous allons aborder le sujet de l'étiquetage des composantes connexes, qui permet d'identifier et de classifier les différentes régions connectées d'une image.

Dans ce rapport, nous allons étudier plusieurs algorithmes pour répondre aux questions posées dans le TP *Opérateurs Connexes et Segmentation*. Nous aborderons notamment les notions d'adjacence et de composantes connexes, en mettant en œuvre différentes méthodes pour l'étiquetage et le filtrage des objets dans une image.

L'objectif de ce rapport est d'analyser ces différentes approches, de comparer leurs performances et d'examiner leurs avantages et inconvénients en fonction du contexte d'utilisation.

L'implémentation du code source associé à ce travail est disponible sur GitHub à l'adresse suivante : **GitHub - Code source du projet**.

Ce dépôt contient l'ensemble des fichiers et des fonctions implémentées, notamment l'étiquetage des composantes connexes, le filtre d'aire et l'optimisation en deux passes.

2 Théorie des Opérateurs Connexes et Segmentation

L'analyse d'images repose sur les relations de voisinage entre pixels, ce qui permet d'identifier des objets, d'effectuer une segmentation et d'améliorer la détection des contours. Dans cette section, nous abordons les notions d'adjacence et de pavage, qui permettent de structurer les pixels sous forme de graphes.

2.1 Adjacence et Relations de Voisinage

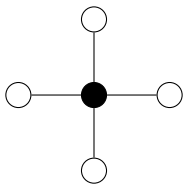
L'analyse d'images repose sur les relations de voisinage entre pixels, qui permettent d'identifier des objets, de segmenter une image et d'améliorer la détection des contours. Nous nous intéresserons ici aux relations de voisinage entre les pixels et aux traitements qui exploitent principalement cette information.
Nous pouvons représenter Une image numérique comme une grille de pixels carrés ayant des coordonnées entières dans Z^2. Cette organisation correspond à un pavage, où chaque pixel est un élément de la grille et possède un certain nombre de voisins définis selon un modèle d'adjacence.

2.2 Adjacence et Relations de Voisinage

Une image numérique est représentée par une grille de pixels carrés ayant des coordonnées entières dans Z^2. Chaque pixel peut avoir plusieurs voisins selon deux types d'adjacence :

— **4-voisinage (N4)** : Un pixel est connecté uniquement à ses 4 voisins directs situés horizontalement et verticalement (haut, bas, gauche, droite). Ce voisinage est défini par :

$$N_4(x, y) = \{(i, j) \in Z^2 \mid |x - i| + |y - j| = 1\} \tag{1}$$

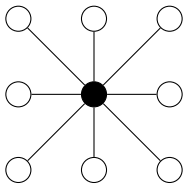


4-voisinage

FIGURE 1 – Illustration du 4-voisinage .

— **8-voisinage (N8)** : Un pixel est connecté à ses 4 voisins directs ainsi qu'aux 4 voisins diagonaux. Ce type d'adjacence permet une connectivité plus complète et est défini par :

$$N_8(x, y) = \{(i, j) \in Z^2 \mid \max(|x - i|, |y - j|) = 1\} \tag{2}$$



8-voisinage

FIGURE 2 – Illustration du 8-voisinage.

L'adjacence permet d'identifier des groupes de pixels connexes formant des objets distincts.
En effet, un changement de voisinage peut avoir un impact sur la détection des objets : une segmentation basée sur le 4-voisinage aura tendance à séparer plus nettement certaines formes, tandis que le 8-voisinage facilitera la connectivité des structures obliques ou diagonales.

On peut présenter l'image comme un graphe, où chaque pixel représente un nœud et les relations de voisinage définissent les arêtes du graphe. Cela permet d'utiliser des algorithmes de graphes et les deux modèles principaux sont :

1. **Graphe de 4-adjacence** : utile pour la segmentation des formes carrées ou rectangulaires.
2. **Graphe de 8-adjacence** : permet de relier plus efficacement les formes diagonales.

3 Résultats de l'Étiquetage des Composantes Connexes

Soit une image binaire, l'objectif de l'algorithme d'étiquetage est de segmenter en identifiant ses composantes connexes selon un critère d'adjacence. Ici, nous devons utiliser la **4-adjacence**.

3.1 Principe de l'Algorithme

L'algorithme explore les pixels de l'image et attribue un label unique à chaque région connexe. Il utilise une exploration en profondeur DFS pour propager le label aux pixels voisins.

Soit une image binaire $I(x, y)$, définie sur un domaine discret $\Omega \subset Z^2$, où :

$$I(x,y) = \begin{cases} 1 & \text{si le pixel appartient à un objet} \\ 0 & \text{sinon} \end{cases}$$

Nous définissons la relation de 4-adjacence $N_4(x, y)$ comme suit :

$$N_4(x,y) = \{(i,j) \in Z^2 \mid |x-i| + |y-j| = 1\}$$

L'algorithme attribue un label unique L_k à chaque composante connexe C_k, où :

$$C_k = \{(x,y) \in \Omega \mid I(x,y) = 1, \text{et il existe un chemin } P \text{ reliant } (x,y) \text{ à un autre pixel de } C_k \text{ dans } N_4\}$$

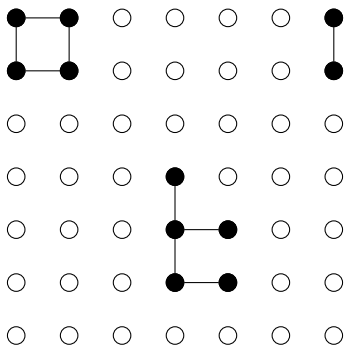


FIGURE 3 – Représentation d'une image 7×7 avec trois composantes connexes en 4-voisinage.

3.2 Exemple d'Application

Représentons la figure 3 sous forme de matrice de taille 7×7, contenant plusieurs composantes connexes :

$$I = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Après application de l'algorithme d'étiquetage des composantes connexes, on obtient :

$$L = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 2 \\ 1 & 1 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 3 & 0 & 0 \\ 0 & 0 & 0 & 3 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Les pixels appartenant à une même composante connexe se voient attribuer un label identique.
Dans cet exemple :

— La première composante (L_1) correspond à l'ensemble des pixels connectés en haut à gauche.
— La deuxième composante (L_2) correspond aux pixels isolés en haut à droite.
— La troisième composante (L_3) représente une région distincte au centre de l'image.

L'algorithme que nous avons implémenter réussit à segmenter les différentes composantes de l'image en fonction du voisinage 4-adjacent.

4 Optimisation de l'Étiquetage des Composantes Connexes : Algorithme en Deux Passes

Bien que l'approche par exploration en profondeur (DFS) soit efficace, il existe une amélioration qui permet d'accélerer l'exécution : l'algorithme d'étiquetage en deux passes qui lui traite l'image ligne par ligne, améliorant ainsi la gestion de la mémoire cache.
L'étiquetage des composantes connexes est un algorithme utilisé en vision par ordinateur permettant d'assigner un label unique à chaque région connectée d'une image binaire.

4.1 Implémentation de l'Algorithme en Deux Passes

L'algorithme en deux passes fonctionne en deux phases principales :

1. **Première passe : Attribution des labels et création d'une table d'équivalence.**

   L'image est donc parcourue ligne par ligne. Chaque pixel de premier plan reçoit un label en fonction de ses voisins :
   — **Aucun voisin étiqueté** : un nouveau label est attribué.
   — **Un seul voisin étiqueté** : le pixel prend son même label.
   — **Plusieurs voisins étiquetés** : le plus petit label est choisi et la table d'équivalence est mise à jour pour fusionner les labels.

2. **Deuxième passe : Fusion des labels via la table d'équivalence.**

   Chaque pixel est remplacé par son label canonique minimal, après la résolution complète de la table d'équivalence.
   -> garantit que tous les pixels appartenant à une même composante connectée portent désormais un label unique.

4.2 Formulation Mathématique

Soit une image binaire $I(x, y)$, définie par :

$$I(x,y) = \begin{cases} 1 & \text{si le pixel appartient à une région d'intérêt} \\ 0 & \text{sinon} \end{cases}$$

Nous définissons le voisinage 4-adjacent :

$$N_4(x,y) = \{(x-1,y), (x,y-1)\}$$

où seuls les voisins de gauche et du haut sont considérés.
L'algorithme suit alors ces règles :

— **Si aucun voisin n'a de label**, un nouveau label est créé.
— **Si un seul voisin est déjà étiqueté**, le pixel prend le même label.
— **Si plusieurs voisins ont des labels différents**, on assigne le plus petit label et on met à jour une table d'équivalence.

4.3 Comparaison entre DFS et l'Algorithme en Deux Passes

L'algorithme en deux passes offre plusieurs avantages comparé à l'approche par exploration en profondeur (DFS). Voici un tableau comparatif :

Critère	DFS	Deux Passes
Méthode	Récursif / pile	Deux étapes
Parcours	Non séquentiel	Séquentiel
Mémoire	Accès irrégulier	Accès optimisé
Attribution	Immédiate	Temporaire, correction
Fusion	Pendant parcours	Table d'équivalence
Complexité	$O(n)$, inefficace	$O(n)$, optimisé
Usage	Petites images	Grandes images

TABLE 1 – Comparaison DFS vs. algorithme en deux passes.

4.4 Optimisation : Gestion des Trous et Structure Hiérarchique

L'algorithme en deux passes fonctionne, mais nous avons essayé d'utiliser une approche similaire pour améliorer la gestion des trous et structurer les composantes connexes.
Nous avons ajouté une étape de détection des trous en identifiant les régions de fond complètement entourées de pixels de premier plan.
Un graphe d'adjacence a été construit pour organiser les relations hiérarchiques entre les différentes composantes détectées.

Voici comment fonctionne l'algorithme :

1. Image Binaire :
— L'image de départ est une grille de pixels binaires.
— Chaque composante est séparée visuellement.

2. Attribution des Labels :
— Chaque groupe de pixels connectés reçoit un label temporaire unique.
— Une table d'équivalence est créée pour fusionner les labels qui sont connectés.
— Dans l'image Labeled Image, chaque élément (yeux, nez, bouche...) a un label de couleur différente.

3. Arbre d'adjacence :
— Un graphe d'équivalence est construit pour associer les labels entre eux.
— Chaque élément de l'image (pupilles, yeux, nez, bouche) est relié à sa catégorie principale.
— Le visage lui-même est connecté au fond blanc pour indiquer qu'il est séparé de l'arrière-plan.

4. Trous remplis :
— Chaque pixel est remplacé par son plus petit label équivalent pour regrouper les composantes.
— Tous les trous sont remplis, et l'image devient un objet entièrement connecté.
— Le fond blanc ne fait plus partie de l'objet, mais le visage est une seule entité unifiée.

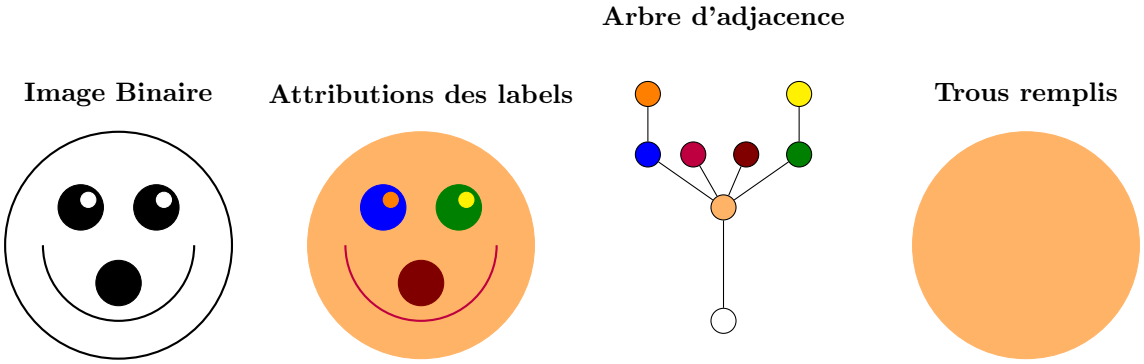


FIGURE 4 – Processus d'étiquetage des composantes connexes avec l'algorithme en deux passes.

5 Filtrage d'Aire des Composantes Connexes

5.1 Objectif

Lorsqu'une image est segmentée, elle contient souvent des petites régions isolées correspondant à du bruit ou à des objets sans importance. Le filtrage d'aire est une technique qui va permettre de supprimer les petites composantes connexes en fonction d'un seuil de taille donné.
Ce processus est essentiel pour :

— Nettoyer une image en éliminant les artefacts ou petites zones isolées.
— Conserver seulement les objets d'intérêt.
— Améliorer la lisibilité de l'image segmentée.

5.2 Définition Mathématique du Filtrage d'Aire

Soit une image binaire $I(x, y)$, définie :

$$I(x,y) = \begin{cases} 1 & \text{si le pixel appartient à une région d'intérêt} \\ 0 & \text{sinon} \end{cases}$$

Il faut identifier les composantes connexes selon un voisinage qu'on a défini (4-voisinage ou 8-voisinage).

Une composante connexe C_i est un sous-ensemble maximal de l'image I tel que pour tout $(x, y) \in C_i$, il existe un chemin de pixels connectés reliant chaque point de C_i.

On définit alors un filtre d'aire F qui supprime toute composante connexe C_i dont la taille $|C_i|$ est inférieure à un seuil S :

$$F(I, S) = \{C_i \in I \mid |C_i| \geq S\}$$

où S est le paramètre déterminant le seuil minimal de pixels à conserver.

5.3 Exemple d'Application

Prenons une image binaire de taille 7×7 contenant trois composantes connexes :

$$I = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 2 \\ 1 & 1 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 3 & 0 & 0 \\ 0 & 0 & 0 & 3 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Si nous appliquons un filtrage d'aire avec un seuil de 3 pixels, nous supprimons la composante 2, car elle contient seulement 2 pixels. Les composantes 1 et 3 sont conservées :

$$F(I,3) = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 3 & 0 & 0 \\ 0 & 0 & 0 & 3 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

5.4 Choix du Type de Label

Deux approches existent après l'application du filtre Soit :

— **Conserver les labels d'origine :** Chaque composante conservée garde son label unique $(1, 3, \ldots)$. -> Permet de suivre chaque région individuellement.

— **Transformer en image binaire** : Utile pour réduire le bruit et simplifier l'image pour des prochains traitements .

Le choix dépend de l'objectif de l'analyse. Si l'on souhaite compter et différencier des objets distincts, il vaut mieux conserver les labels. Si l'objectif est d'améliorer l'image pour d'autres transformations, une image binaire est plus adaptée.

5.5 Conclusion

Le filtrage d'aire est une méthode efficace pour éliminer les petites structures non pertinentes et se concentrer sur les objets significatifs. Son intégration dans un pipeline de traitement d'image permet d'améliorer la*qualité des données avant d'appliquer des analyses plus avancées. Son efficacité dépend du seuil de taille S choisi.

6 Conclusion

En conclusion nous pouvons dire que nous avons exploré plusieurs méthodes d'étiquetage et de filtrage des composantes connexes dans une image binaire. Nous avons étudié particulièrement l'adjacence et les relations de voisinage, importantes pour structurer les pixels sous forme de graphes et permettre une segmentation efficace des objets.
Dans un premier temps, nous avons implémenté un algorithme d'étiquetage en composantes connexes en utilisant une approche basée sur l'exploration en profondeur ou en largeur. Nous avons vu que cette méthode a montré ses limites en termes d'accès mémoireet de performances limitées pour des images de grande taille. Pour pallier ces limitations, nous avons ensuite étudié et implémenté une optimisation en deux passes. Cette approche a permis une meilleure gestion de la mémoire cache en parcourant l'image de manière séquentielle. Nous avons également introduit une amélioration supplémentaire avec une gestion des trous et une structure hiérarchique des composantes connexes, permettant une segmentation plus précise.
Pour finir nous avons abordé le filtrage d'aire. Cette technique est utile pour éliminer le bruit et améliorer la qualité de la segmentation en conservant uniquement les objets d'intérêt.

En conclusion, ces différentes approches nous ont montré comment l'étiquetage des composantes connexes peut être amélioré et adapté selon les besoins spécifiques d'une application.