

Partitionnement d'image robuste pour l'indexation par région

Olivier Chekroun

5 juillet 2004

Introduction

Objectifs

Algorithmes de segmentation

Basé sur un graphe : MInt et Segment

K-Means

Mean Shift

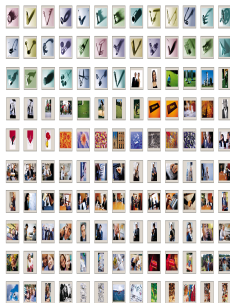
Mise en oeuvre

Structure de graphe hiérarchique

Labellisation et recherche de frontières

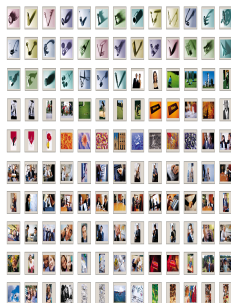
Images de sortie

Introduction



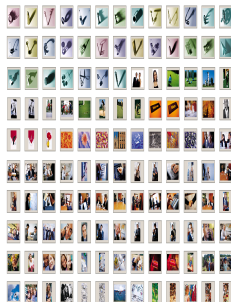
- Prolifération des collections d'images
- Besoin d'effectuer des recherches dans ces collections
- Indexation des images
- Approche basée sur les régions : on essaye d'extraire des images des régions représentant des objets

Introduction



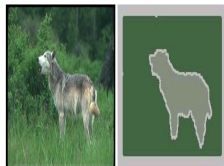
- Prolifération des collections d'images
- Besoin d'effectuer des recherches dans ces collections
- Indexation des images
- Approche basée sur les régions : on essaye d'extraire des images des régions représentant des objets

Introduction



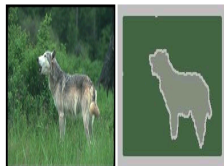
- Prolifération des collections d'images
- Besoin d'effectuer des recherches dans ces collections
- Indexation des images
- Approche basée sur les régions : on essaye d'extraire des images des régions représentant des objets

Partitionnement



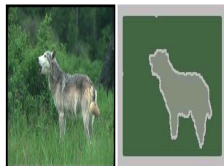
- Segmentation ou partitionnement : manière d'extraire les objets d'une image
- La recherche d'objets dans une image est un problème complexe et souvent mal posé
- Les algorithmes sont démontrés en illustrant leurs résultats sur des images choisies et avec des paramètres précis
- On veut des résultats homogènes sur une collection d'images

Partitionnement



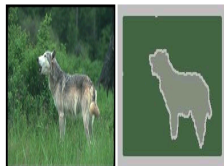
- Segmentation ou partitionnement : manière d'extraire les objets d'une image
- La recherche d'objets dans une image est un problème complexe et souvent mal posé
- Les algorithmes sont démontrés en illustrant leurs résultats sur des images choisies et avec des paramètres précis
- On veut des résultats homogènes sur une collection d'images

Partitionnement



- Segmentation ou partitionnement : manière d'extraire les objets d'une image
- La recherche d'objets dans une image est un problème complexe et souvent mal posé
- Les algorithmes sont démontrés en illustrant leurs résultats sur des images choisies et avec des paramètres précis
- On veut des résultats homogènes sur une collection d'images

Partitionnement



- Segmentation ou partitionnement : manière d'extraire les objets d'une image
- La recherche d'objets dans une image est un problème complexe et souvent mal posé
- Les algorithmes sont démontrés en illustrant leurs résultats sur des images choisies et avec des paramètres précis
- On veut des résultats homogènes sur une collection d'images

Objectifs

- Pas une nouvelle méthode de segmentation miracle mais utilisation d'algorithmes établis.
- Fournir un programme de partitionnement d'image robuste
- Le programme doit donner un résultat permettant l'indexation
- Implémentation en C++ : lisibilité et portabilité
- Format d'image PNG, pour stocker le résultat sans perte
- Le programme doit implémenter 3 algorithmes de segmentation spécifiques :
 -
 -
 -

Objectifs

- Pas une nouvelle méthode de segmentation miracle mais utilisation d'algorithmes établis.
- Fournir un programme de partitionnement d'image robuste
- Le programme doit donner un résultat permettant l'indexation
- Implémentation en C++ : lisibilité et portabilité
- Format d'image PNG, pour stocker le résultat sans perte
- Le programme doit implémenter 3 algorithmes de segmentation spécifiques :
 -
 -
 -

Objectifs

- Pas une nouvelle méthode de segmentation miracle mais utilisation d'algorithmes établis.
- Fournir un programme de partitionnement d'image robuste
- Le programme doit donner un résultat permettant l'indexation
- Implémentation en C++ : lisibilité et portabilité
- Format d'image PNG, pour stocker le résultat sans perte
- Le programme doit implémenter 3 algorithmes de segmentation spécifiques :
 -
 -
 -

Objectifs

- Pas une nouvelle méthode de segmentation miracle mais utilisation d'algorithmes établis.
- Fournir un programme de partitionnement d'image robuste
- Le programme doit donner un résultat permettant l'indexation
- Implémentation en C++ : lisibilité et portabilité
- Format d'image PNG, pour stocker le résultat sans perte
- Le programme doit implémenter 3 algorithmes de segmentation spécifiques :
 - Méthode de segmentation par régions
 - Méthode de segmentation par contours
 - Méthode de segmentation par pixels

Objectifs

- Pas une nouvelle méthode de segmentation miracle mais utilisation d'algorithmes établis.
- Fournir un programme de partitionnement d'image robuste
- Le programme doit donner un résultat permettant l'indexation
- Implémentation en C++ : lisibilité et portabilité
- Format d'image PNG, pour stocker le résultat sans perte
- Le programme doit implémenter 3 algorithmes de segmentation spécifiques :
 - "MInt" ou "Basé sur un graphe" et une version modifiée
 - L'algorithme des K-Means
 - L'algorithme du "Mean Shift"

Objectifs

- Pas une nouvelle méthode de segmentation miracle mais utilisation d'algorithmes établis.
- Fournir un programme de partitionnement d'image robuste
- Le programme doit donner un résultat permettant l'indexation
- Implémentation en C++ : lisibilité et portabilité
- Format d'image PNG, pour stocker le résultat sans perte
- Le programme doit implémenter 3 algorithmes de segmentation spécifiques :
 - "MInt" ou "Basé sur un graphe" et une version modifiée
 - L'algorithme des K-Means
 - L'algorithme du "Mean Shift"

Objectifs

- Pas une nouvelle méthode de segmentation miracle mais utilisation d'algorithmes établis.
- Fournir un programme de partitionnement d'image robuste
- Le programme doit donner un résultat permettant l'indexation
- Implémentation en C++ : lisibilité et portabilité
- Format d'image PNG, pour stocker le résultat sans perte
- Le programme doit implémenter 3 algorithmes de segmentation spécifiques :
 - "MInt" ou "Basé sur un graphe" et une version modifiée
 - L'algorithme des K-Means
 - L'algorithme du "Mean Shift"

Algorithme basé sur un graphe

On a un graphe $G = (V, E)$:

- les sommets sont les pixels
- les arêtes sont des arêtes d'adjacence entre pixel voisin et leur poids la distance dans l'espace de couleurs des pixels reliés par ces arêtes

Par rapport à ce graphe, on utilise :

- structure d'arbre de recouvrement minimal
- algorithme de Kruskal
- prédicat de fusion de région

Algorithme basé sur un graphe

On a un graphe $G = (V, E)$:

- les sommets sont les pixels
- les arêtes sont des arêtes d'adjacence entre pixel voisin et leur poids la distance dans l'espace de couleurs des pixels reliés par ces arêtes

Par rapport à ce graphe, on utilise :

- structure d'arbre de recouvrement minimal
- algorithme de Kruskal
- prédicat de fusion de région

Propriétés du prédicat

- S'adapte pendant le déroulement de l'algorithme
- En ne prenant que des décisions locales on obtient des propriétés globales
- Le prédicat mesure la présence d'une frontière entre 2 régions en comparant 2 quantités :
 - la variation à travers la frontière
 - la variation entre points voisins d'une région

Propriétés du prédicat

- S'adapte pendant le déroulement de l'algorithme
- En ne prenant que des décisions locales on obtient des propriétés globales
- Le prédicat mesure la présence d'une frontière entre 2 régions en comparant 2 quantités :
 - la variation à travers la frontière
 - la variation entre points voisins d'une région

Prédicat de fusion

- Variation interne : $Int(C) = \max_{e \in MST(C,E)} w(e)$
- Tolérance : $\tau(C) = \frac{k}{|C|}$
- Variation minimal :
 $MInt(C_1, C_2) = \min(Int(C_1) + \tau(C_1), Int(C_2) + \tau(C_2))$
- Variation externe :
 $Dif(C_1, C_2) = \min_{v_i \in C_1, v_j \in C_2, (v_i, v_j) \in E} w(v_i, v_j)$
- Prédicat :

$$D(C_1, C_2) = \begin{cases} vrai & \text{si } Dif(C_1, C_2) > MInt(C_1, C_2) \\ faux & \text{sinon} \end{cases}$$

Prédicat de fusion

- Variation interne : $Int(C) = \max_{e \in MST(C,E)} w(e)$
- Tolérance : $\tau(C) = \frac{k}{|C|}$
- Variation minimal :
 $MInt(C_1, C_2) = \min(Int(C_1) + \tau(C_1), Int(C_2) + \tau(C_2))$
- Variation externe :
 $Dif(C_1, C_2) = \min_{v_i \in C_1, v_j \in C_2, (v_i, v_j) \in E} w(v_i, v_j)$
- Prédicat :

$$D(C_1, C_2) = \begin{cases} vrai & \text{si } Dif(C_1, C_2) > MInt(C_1, C_2) \\ faux & \text{sinon} \end{cases}$$

Prédicat de fusion

- Variation interne : $Int(C) = \max_{e \in MST(C,E)} w(e)$
- Tolérance : $\tau(C) = \frac{k}{|C|}$
- Variation minimal :
 $MInt(C_1, C_2) = \min(Int(C_1) + \tau(C_1), Int(C_2) + \tau(C_2))$
- Variation externe :
 $Dif(C_1, C_2) = \min_{v_i \in C_1, v_j \in C_2, (v_i, v_j) \in E} w(v_i, v_j)$
- Prédicat :

$$D(C_1, C_2) = \begin{cases} vrai & \text{si } Dif(C_1, C_2) > MInt(C_1, C_2) \\ faux & \text{sinon} \end{cases}$$

Description de l'algorithme

Soit un graphe $G = (V, E)$, avec n sommets et m arrêtes.

- ① Tri de E par poids non décroissant.
- ② On commence par une segmentation S^0 où chaque sommet v_i est sa propre régions.
- ③ Pour $q = 1 \dots m$, on construit S^q à partir de S^{q-1} comme suit. Soit v_i et v_j les sommets connectés par la q -ième arrête, C_i^{q-1} la région de S^{q-1} contenant v_i et C_j^{q-1} la région de S^{q-1} contenant v_j . Si $C_i^{q-1} \neq C_j^{q-1}$ et $w(o_q) \leq MInt(C_i^{q-1}, C_j^{q-1})$, alors on fusionne C_i^{q-1} et C_j^{q-1} pour obtenir S^q , sinon $S^q = S^{q-1}$.

Description de l'algorithme

Soit un graphe $G = (V, E)$, avec n sommets et m arrêtes.

- 1 Tri de E par poids non décroissant.
- 2 On commence par une segmentation S^0 où chaque sommet v_i est sa propre régions.
- 3 Pour $q = 1 \dots m$, on construit S^q à partir de de S^{q-1} comme suit. Soit v_i et v_j les sommets connectés par la q -ième arrête, C_i^{q-1} la région de S^{q-1} contenant v_i et C_j^{q-1} la région de S^{q-1} contenant v_j . Si $C_i^{q-1} \neq C_j^{q-1}$ et $w(o_q) \leq MInt(C_i^{q-1}, C_j^{q-1})$, alors on fusionne C_i^{q-1} et C_j^{q-1} pour obtenir S^q , sinon $S^q = S^{q-1}$.

Description de l'algorithme

Soit un graphe $G = (V, E)$, avec n sommets et m arrêtes.

- ① Tri de E par poids non décroissant.
- ② On commence par une segmentation S^0 où chaque sommet v_i est sa propre régions.
- ③ Pour $q = 1 \dots m$, on construit S^q à partir de de S^{q-1} comme suit. Soit v_i et v_j les sommets connectés par la q -ième arrête, C_i^{q-1} la région de S^{q-1} contenant v_i et C_j^{q-1} la région de S^{q-1} contenant v_j . Si $C_i^{q-1} \neq C_j^{q-1}$ et $w(o_q) \leq MInt(C_i^{q-1}, C_j^{q-1})$, alors on fusionne C_i^{q-1} et C_j^{q-1} pour obtenir S^q , sinon $S^q = S^{q-1}$.

Description de l'algorithme

Soit un graphe $G = (V, E)$, avec n sommets et m arrêtes.

- ❶ Tri de E par poids non décroissant.
- ❷ On commence par une segmentation S^0 où chaque sommet v_i est sa propre régions.
- ❸ Pour $q = 1 \dots m$, on construit S^q à partir de S^{q-1} comme suit. Soit v_i et v_j les sommets connectés par la q -ième arrête, C_i^{q-1} la région de S^{q-1} contenant v_i et C_j^{q-1} la région de S^{q-1} contenant v_j . Si $C_i^{q-1} \neq C_j^{q-1}$ et $w(o_q) \leq MInt(C_i^{q-1}, C_j^{q-1})$, alors on fusionne C_i^{q-1} et C_j^{q-1} pour obtenir S^q , sinon $S^q = S^{q-1}$.

Description de l'algorithme

Soit un graphe $G = (V, E)$, avec n sommets et m arrêtes.

- ❶ Tri de E par poids non décroissant.
- ❷ On commence par une segmentation S^0 où chaque sommet v_i est sa propre régions.
- ❸ Pour $q = 1 \dots m$, on construit S^q à partir de de S^{q-1} comme suit. Soit v_i et v_j les sommets connectés par la q -ième arrête, C_i^{q-1} la région de S^{q-1} contenant v_i et C_j^{q-1} la région de S^{q-1} contenant v_j . Si $C_i^{q-1} \neq C_j^{q-1}$ et $w(o_q) \leq MInt(C_i^{q-1}, C_j^{q-1})$, alors on fusionne C_i^{q-1} et C_j^{q-1} pour obtenir S^q , sinon $S^q = S^{q-1}$.

Problèmes rencontrés

Bien que relativement intuitif cet algorithme laisse quelques points en suspens.

- On n'a pas une définition précise du poids des arêtes.
- Que représente la valeur de τ ?
- Le paramètre ne s'adapte pas aux images : pas robuste
- Pas d'évaluation de la qualité d'une fusion.

Prédicat segment

Doit on fusionner les composants C_1 et C_2 s'il sont reliés par l'arête e ?

- Normalisation : $k' = k\Delta$ où Δ est la distance maximum de l'image
- Calcule de : $k_1 = \frac{k'}{|C_1|}$ et $k_2 = \frac{k'}{|C_2|}$
- Poids moyens de chaque région : $p_1 = \frac{1}{|C_1|} \sum_{p \in C_1}$ et $p_2 = \frac{1}{|C_2|} \sum_{p \in C_2}$
- Tolérance (prise en compte de la taille) : $f_1 = p_1 + k_1$ et $f_2 = p_2 + k_2$
- Estimation : $estimation = \frac{f_1 + f_2 + w(e)}{3}$
- Prédicat :

$$D(C_1, C_2) = \begin{cases} vrai & \text{si } w(e) \leq estimation \\ faux & \text{sinon} \end{cases}$$

Prédicat segment

Doit on fusionner les composants C_1 et C_2 s'il sont reliés par l'arête e ?

- Normalisation : $k' = k\Delta$ où Δ est la distance maximum de l'image
- Calcule de : $k_1 = \frac{k'}{|C_1|}$ et $k_2 = \frac{k'}{|C_2|}$
- Poids moyens de chaque région : $p_1 = \frac{1}{|C_1|} \sum_{p \in C_1}$ et $p_2 = \frac{1}{|C_2|} \sum_{p \in C_2}$
- Tolérance (prise en compte de la taille) : $f_1 = p_1 + k_1$ et $f_2 = p_2 + k_2$
- Estimation : $estimation = \frac{f_1 + f_2 + w(e)}{3}$
- Prédicat :

$$D(C_1, C_2) = \begin{cases} vrai & \text{si } w(e) \leq estimation \\ faux & \text{sinon} \end{cases}$$

Exemple de segmentation basée sur un graphe



3 régions trouvées :

- la rampe dégradée
- le rectangle de droite
- la zone de "turbulences"

C'est le meilleur résultats sur cette image synthétique.

Déroulement de l'algorithme basé sur un graphe

Algorithme des K-Means

- Algorithme de classification beaucoup employé car très simple
- Répartir l'ensemble des points d'un espace autour de k points représentatifs
- Segmentation d'image : espace des couleurs des pixels.
- Paléttisation les k centres sont les couleurs représentatives de l'ensemble de l'image.

Algorithme des K-Means

- Algorithme de classification beaucoup employé car très simple
- Répartir l'ensemble des points d'un espace autour de k points représentatifs
- Segmentation d'image : espace des couleurs des pixels.
- Paléttisation les k centres sont les couleurs représentatives de l'ensemble de l'image.

Description de l'algorithme

Soient des points p_j , k un nombre de centres et L_i l'étiquette du i -ème pixel après application de l'algorithme.

- ① On choisit $K_{i=1\dots k}$ centres
- ② On répète jusqu'à convergence.
 - Affectation chaque p_j au centre le plus proche K_i .
 - Calcul de la moyenne de chaque centre $m_i = \frac{1}{|K_i|} \sum_{p \in K_i}$
 - Affectation des nouveaux centre $K_i = m_i$
- ③ Pour chaque $p_{j=1\dots n}$, on affecte l'étiquette : $L_i = \{p \in K_i\}$

2 difficultés avec cet algorithme : le nombre de centre k et les couleurs initiales.

Description de l'algorithme

Soient des points p_j , k un nombre de centres et L_i l'étiquette du i -ème pixel après application de l'algorithme.

- 1 On choisit $K_{i=1\dots k}$ centres
- 2 On répète jusqu'à convergence.
 - Affectation chaque p_j au centre le plus proche K_i .
 - Calcul de la moyenne de chaque centre $m_i = \frac{1}{|K_i|} \sum_{p \in K_i}$
 - Affectation des nouveaux centre $K_i = m_i$
- 3 Pour chaque $p_{j=1\dots n}$, on affecte l'étiquette : $L_i = \{p \in K_i\}$

2 difficultés avec cet algorithme : le nombre de centre k et les couleurs initiales.

Description de l'algorithme

Soient des points p_j , k un nombre de centres et L_i l'étiquette du i -ème pixel après application de l'algorithme.

- 1 On choisit $K_{i=1\dots k}$ centres
- 2 On répète jusqu'à convergence.
 - Affectation chaque p_j au centre le plus proche K_i .
 - Calcul de la moyenne de chaque centre $m_i = \frac{1}{|K_i|} \sum_{p \in K_i}$
 - Affectation des nouveaux centre $K_i = m_i$
- 3 Pour chaque $p_{j=1\dots n}$, on affecte l'étiquette : $L_i = \{p \in K_i\}$

2 difficultés avec cet algorithme : le nombre de centre k et les couleurs initiales.

Description de l'algorithme

Soient des points p_j , k un nombre de centres et L_i l'étiquette du i -ème pixel après application de l'algorithme.

- ① On choisit $K_{i=1\dots k}$ centres
- ② On répète jusqu'à convergence.
 - Affectation chaque p_j au centre le plus proche K_i .
 - Calcul de la moyenne de chaque centre $m_i = \frac{1}{|K_i|} \sum_{p \in K_i}$
 - Affectation des nouveaux centre $K_i = m_i$
- ③ Pour chaque $p_{j=1\dots n}$, on affecte l'étiquette : $L_i = \{p \in K_i\}$

2 difficultés avec cet algorithme : le nombre de centre k et les couleurs initiales.

Description de l'algorithme

Soient des points p_j , k un nombre de centres et L_i l'étiquette du i -ème pixel après application de l'algorithme.

- ① On choisit $K_{i=1\dots k}$ centres
- ② On répète jusqu'à convergence.
 - Affectation chaque p_j au centre le plus proche K_i .
 - Calcul de la moyenne de chaque centre $m_i = \frac{1}{|K_i|} \sum_{p \in K_i}$
 - Affectation des nouveaux centre $K_i = m_i$
- ③ Pour chaque $p_{j=1\dots n}$, on affecte l'étiquette : $L_i = \{p \in K_i\}$

2 difficultés avec cet algorithme : le nombre de centre k et les couleurs initiales.

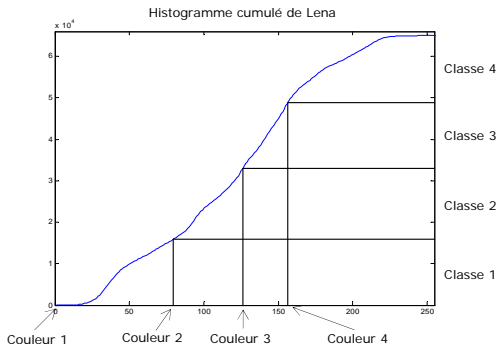
Description de l'algorithme

Soient des points p_j , k un nombre de centres et L_i l'étiquette du i -ème pixel après application de l'algorithme.

- ① On choisit $K_{i=1\dots k}$ centres
- ② On répète jusqu'à convergence.
 - Affectation chaque p_j au centre le plus proche K_i .
 - Calcul de la moyenne de chaque centre $m_i = \frac{1}{|K_i|} \sum_{p \in K_i}$
 - Affectation des nouveaux centre $K_i = m_i$
- ③ Pour chaque $p_{j=1\dots n}$, on affecte l'étiquette : $L_i = \{p \in K_i\}$

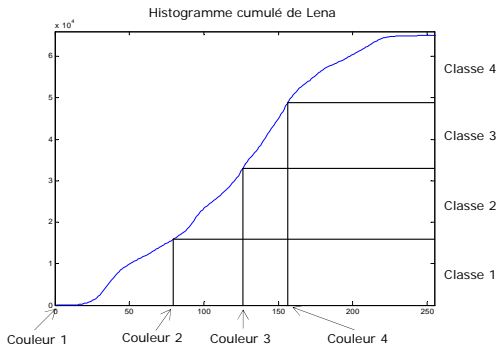
2 difficultés avec cet algorithme : le nombre de centre k et les couleurs initiales.

Choix des couleurs initiales



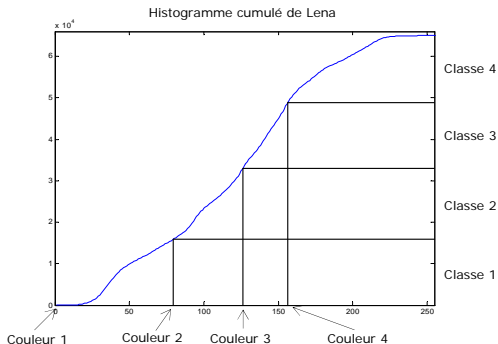
- Calcule de l'histogramme cumulé de l'image
- Division de l'axe des cardinaux en k classes
- On choisit un point dans l'image inverse des cardinaux

Choix des couleurs initiales



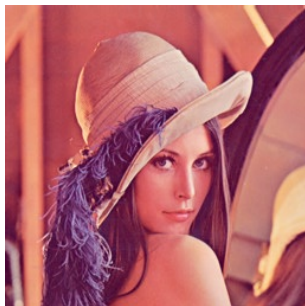
- Calcule de l'histogramme cumulé de l'image
- Division de l'axe des cardinaux en k classes
- On choisit un point dans l'image inverse des cardinaux

Choix des couleurs initiales



- Calcule de l'histogramme cumulé de l'image
- Division de l'axe des cardinaux en k classes
- On choisit un point dans l'image inverse des cardinaux

Exemple de segmentation par la méthode des K-Means



- Paramètre : 8
- Résultat : 888 composants et 2310 arêtes

Méthode du Mean Shift

- La procédure du Mean Shift est une méthode récente d'analyse de données pour en trouver les modes.
- Englobe plusieurs algorithmes de segmentation.
- Principe : calculer le gradient d'une fenêtre d'exploration des données centrée en un point. Ce vecteur indique la direction d'un mode local.
- Approximer ce vecteur en calculant la moyenne des points de la fenêtre.
- Déplacement du centre de la fenêtre au point calculé et en itérant ce calcul sur le nouveau point calculé on converge rapidement vers un mode local.

Méthode du Mean Shift

- La procédure du Mean Shift est une méthode récente d'analyse de données pour en trouver les modes.
- Englobe plusieurs algorithmes de segmentation.
- Principe : calculer le gradient d'une fenêtre d'exploration des données centrée en un point. Ce vecteur indique la direction d'un mode local.
- Approximer ce vecteur en calculant la moyenne des points de la fenêtre.
- Déplacement du centre de la fenêtre au point calculé et en itérant ce calcul sur le nouveau point calculé on converge rapidement vers un mode local.

Méthode du Mean Shift

- La procédure du Mean Shift est une méthode récente d'analyse de données pour en trouver les modes.
- Englobe plusieurs algorithmes de segmentation.
- Principe : calculer le gradient d'une fenêtre d'exploration des données centrée en un point. Ce vecteur indique la direction d'un mode local.
- Approximer ce vecteur en calculant la moyenne des points de la fenêtre.
- Déplacement du centre de la fenêtre au point calculé et en itérant ce calcul sur le nouveau point calculé on converge rapidement vers un mode local.

Méthode du Mean Shift

- La procédure du Mean Shift est une méthode récente d'analyse de données pour en trouver les modes.
- Englobe plusieurs algorithmes de segmentation.
- Principe : calculer le gradient d'une fenêtre d'exploration des données centrée en un point. Ce vecteur indique la direction d'un mode local.
- Approximer ce vecteur en calculant la moyenne des points de la fenêtre.
- Déplacement du centre de la fenêtre au point calculé et en itérant ce calcul sur le nouveau point calculé on converge rapidement vers un mode local.

Méthode du Mean Shift

- La procédure du Mean Shift est une méthode récente d'analyse de données pour en trouver les modes.
- Englobe plusieurs algorithmes de segmentation.
- Principe : calculer le gradient d'une fenêtre d'exploration des données centrée en un point. Ce vecteur indique la direction d'un mode local.
- Approximer ce vecteur en calculant la moyenne des points de la fenêtre.
- Déplacement du centre de la fenêtre au point calculé et en itérant ce calcul sur le nouveau point calculé on converge rapidement vers un mode local.

Concrètement

- Pour l'analyse d'image on considère l'espace conjoint des données : spatial et attribut.
- 3 coordonnées (x, y, g) avec g le niveau de gris pour une image en niveau de gris
- 5 coordonnées (x, y, h, s, v) avec h, s, v les composants de la couleur
- En pratique on peut largement simplifier :
 - Utilisation du noyau d'Epanechnikov
 - Calcul des modes par une recherche binaire
 - Remplacement des coordonnées spatiales par une seule coordonnée

Concrètement

- Pour l'analyse d'image on considère l'espace conjoint des données : spatial et attribut.
- 3 coordonnées (x, y, g) avec g le niveau de gris pour une image en niveau de gris
- 5 coordonnées (x, y, h, s, v) avec h, s, v les composants de la couleur
- En pratique on peut largement simplifier :
 - Utilisation du noyau d'Epanechnikov
 - Calcul du noyau remplacé par une moyenne
 - Calcul du noyau de couleur remplacé par la moyenne

Concrètement

- Pour l'analyse d'image on considère l'espace conjoint des données : spatial et attribut.
- 3 coordonnées (x, y, g) avec g le niveau de gris pour une image en niveau de gris
- 5 coordonnées (x, y, h, s, v) avec h, s, v les composants de la couleur
- En pratique on peut largement simplifier :
 - Utilisation du noyau d'Epanechnikov
 - Calcul du noyau remplacé par une moyenne
 - Remplacement de la sphère par un "hyper rectangle"

Concrètement

- Pour l'analyse d'image on considère l'espace conjoint des données : spatial et attribut.
- 3 coordonnées (x, y, g) avec g le niveau de gris pour une image en niveau de gris
- 5 coordonnées (x, y, h, s, v) avec h, s, v les composants de la couleur
- En pratique on peut largement simplifier :
 - Utilisation du noyau d'Epanechnikov
 - Calcul du noyau remplacé par une moyenne
 - Remplacement de la sphère par un "hyper rectangle"

Concrètement

- Pour l'analyse d'image on considère l'espace conjoint des données : spatial et attribut.
- 3 coordonnées (x, y, g) avec g le niveau de gris pour une image en niveau de gris
- 5 coordonnées (x, y, h, s, v) avec h, s, v les composants de la couleur
- En pratique on peut largement simplifier :
 - Utilisation du noyau d'Epanechnikov
 - Calcul du noyau remplacé par une moyenne
 - Remplacement de la sphère par un "hyper rectangle"

Concrètement

- Pour l'analyse d'image on considère l'espace conjoint des données : spatial et attribut.
- 3 coordonnées (x, y, g) avec g le niveau de gris pour une image en niveau de gris
- 5 coordonnées (x, y, h, s, v) avec h, s, v les composants de la couleur
- En pratique on peut largement simplifier :
 - Utilisation du noyau d'Epanechnikov
 - Calcul du noyau remplacé par une moyenne
 - Remplacement de la sphère par un "hyper rectangle"

Filtre Mean Shift

Soient $\{x_j\}_{j=1\dots n}$ et $\{z_j\}_{j=1\dots n}$ les points originaux et filtrés de l'image dans le domaine joint spatial-attribut. Les indices s et a représentent respectivement les coordonnées spatiales et d'attributs.

Pour $j = 1 \dots n$

- 1 Initialisation : $k = 1$ et $y_k = x_j$
- 2 Calcul de y_{k+1} par le mean shift jusqu'à convergence : y_c
- 3 $z_j = (x_j^s, y_c^a)$

Filtre Mean Shift

Soient $\{x_j\}_{j=1\dots n}$ et $\{z_j\}_{j=1\dots n}$ les points originaux et filtrés de l'image dans le domaine joint spatial-attribut. Les indices s et a représentent respectivement les coordonnées spatiales et d'attributs.

Pour $j = 1 \dots n$

- 1 Initialisation : $k = 1$ et $y_k = x_j$
- 2 Calcul de y_{k+1} par le mean shift jusqu'à convergence : y_c
- 3 $z_j = (x_j^s, y_c^a)$

Filtre Mean Shift

Soient $\{x_j\}_{j=1\dots n}$ et $\{z_j\}_{j=1\dots n}$ les points originaux et filtrés de l'image dans le domaine joint spatial-attribut. Les indices s et a représentent respectivement les coordonnées spatiales et d'attributs.

Pour $j = 1 \dots n$

- ❶ Initialisation : $k = 1$ et $y_k = x_j$
- ❷ Calcul de y_{k+1} par le mean shift jusqu'à convergence : y_c
- ❸ $z_j = (x_j^s, y_c^a)$

Filtre Mean Shift

Soient $\{x_j\}_{j=1\dots n}$ et $\{z_j\}_{j=1\dots n}$ les points originaux et filtrés de l'image dans le domaine joint spatial-attribut. Les indices s et a représentent respectivement les coordonnées spatiales et d'attributs.

Pour $j = 1 \dots n$

- 1 Initialisation : $k = 1$ et $y_k = x_j$
- 2 Calcul de y_{k+1} par le mean shift jusqu'à convergence : y_c
- 3 $z_j = (x_j^s, y_c^a)$

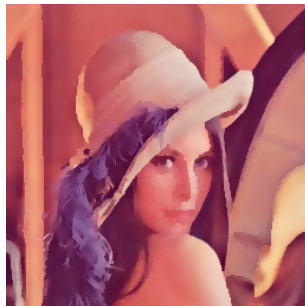
Filtre Mean Shift

Soient $\{x_j\}_{j=1\dots n}$ et $\{z_j\}_{j=1\dots n}$ les points originaux et filtrés de l'image dans le domaine joint spatial-attribut. Les indices s et a représentent respectivement les coordonnées spatiales et d'attributs.

Pour $j = 1 \dots n$

- 1 Initialisation : $k = 1$ et $y_k = x_j$
- 2 Calcul de y_{k+1} par le mean shift jusqu'à convergence : y_c
- 3 $z_j = (x_j^s, y_c^a)$

Exemple de filtre Mean Shift



- Paramètre spatial : 3
- Paramètre dans l'espace des attributs : 0.2
- Résultat : 5522 composants et 215036 arêtes

Segmentation Mean Shift

Soient $\{x_j\}_{j=1\dots n}$ et $\{z_j\}_{j=1\dots n}$ les points originaux et filtrés de l'image dans le domaine joint spatial-attribut, et L_i l'étiquette du i -ème pixel dans l'image segmentée.

- ➊ Application du filtre Mean Shift à l'image, mais $z_i = y_c$
- ➋ Trouver dans le domaine joint les régions $\{C_p\}_{p=1\dots m}$ en groupant tous les z_i qui sont plus proche de h_s dans le domaine spatial et h_a dans le domaine des attributs
- ➌ Pour chaque $j = 1 \dots n$, on affecte l'étiquette :
 $L_i = \{p \mid z_i \in C_p\}$
- ➍ Option : éliminer les régions qui moins de M pixels

Segmentation Mean Shift

Soient $\{x_j\}_{j=1\dots n}$ et $\{z_j\}_{j=1\dots n}$ les points originaux et filtrés de l'image dans le domaine joint spatial-attribut, et L_i l'étiquette du i -ème pixel dans l'image segmentée.

- 1 Application du filtre Mean Shift à l'image, mais $z_i = y_c$
- 2 Trouver dans le domaine joint les régions $\{C_p\}_{p=1\dots m}$ en groupant tous les z_i qui sont plus proche de h_s dans le domaine spatial et h_a dans le domaine des attributs
- 3 Pour chaque $j = 1 \dots n$, on affecte l'étiquette :
 $L_i = \{p \mid z_i \in C_p\}$
- 4 Option : éliminer les régions qui moins de M pixels

Segmentation Mean Shift

Soient $\{x_j\}_{j=1\dots n}$ et $\{z_j\}_{j=1\dots n}$ les points originaux et filtrés de l'image dans le domaine joint spatial-attribut, et L_i l'étiquette du i -ème pixel dans l'image segmentée.

- ➊ Application du filtre Mean Shift à l'image, mais $z_i = y_c$
- ➋ Trouver dans le domaine joint les régions $\{C_p\}_{p=1\dots m}$ en groupant tous les z_i qui sont plus proche de h_s dans le domaine spatial et h_a dans le domaine des attributs
- ➌ Pour chaque $j = 1 \dots n$, on affecte l'étiquette :
 $L_i = \{p \mid z_i \in C_p\}$
- ➍ Option : éliminer les régions qui moins de M pixels

Segmentation Mean Shift

Soient $\{x_j\}_{j=1\dots n}$ et $\{z_j\}_{j=1\dots n}$ les points originaux et filtrés de l'image dans le domaine joint spatial-attribut, et L_i l'étiquette du i -ème pixel dans l'image segmentée.

- ➊ Application du filtre Mean Shift à l'image, mais $z_i = y_c$
- ➋ Trouver dans le domaine joint les régions $\{C_p\}_{p=1\dots m}$ en groupant tous les z_i qui sont plus proche de h_s dans le domaine spatial et h_a dans le domaine des attributs
- ➌ Pour chaque $j = 1 \dots n$, on affecte l'étiquette :
 $L_i = \{p \mid z_i \in C_p\}$
- ➍ Option : éliminer les régions qui moins de M pixels

Segmentation Mean Shift

Soient $\{x_j\}_{j=1\dots n}$ et $\{z_j\}_{j=1\dots n}$ les points originaux et filtrés de l'image dans le domaine joint spatial-attribut, et L_i l'étiquette du i -ème pixel dans l'image segmentée.

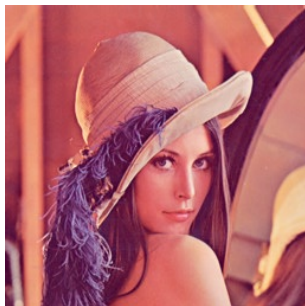
- ➊ Application du filtre Mean Shift à l'image, mais $z_i = y_c$
- ➋ Trouver dans le domaine joint les régions $\{C_p\}_{p=1\dots m}$ en groupant tous les z_i qui sont plus proche de h_s dans le domaine spatial et h_a dans le domaine des attributs
- ➌ Pour chaque $j = 1 \dots n$, on affecte l'étiquette :
$$L_i = \{p \mid z_i \in C_p\}$$
- ➍ Option : éliminer les régions qui moins de M pixels

Segmentation Mean Shift

Soient $\{x_j\}_{j=1\dots n}$ et $\{z_j\}_{j=1\dots n}$ les points originaux et filtrés de l'image dans le domaine joint spatial-attribut, et L_i l'étiquette du i -ème pixel dans l'image segmentée.

- ➊ Application du filtre Mean Shift à l'image, mais $z_i = y_c$
- ➋ Trouver dans le domaine joint les régions $\{C_p\}_{p=1\dots m}$ en groupant tous les z_i qui sont plus proche de h_s dans le domaine spatial et h_a dans le domaine des attributs
- ➌ Pour chaque $j = 1 \dots n$, on affecte l'étiquette :
$$L_i = \{p \mid z_i \in C_p\}$$
- ➍ Option : éliminer les régions qui moins de M pixels

Exemple de segmentation Mean Shift



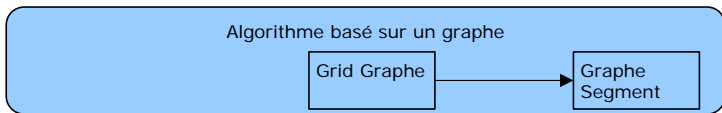
- Paramètre spatial : 3
- Paramètre dans l'espace des attributs : 0.2
- Elagage des régions de moins de 500 pixels
- Résultat : 19 composants et 37 arêtes

Workflow

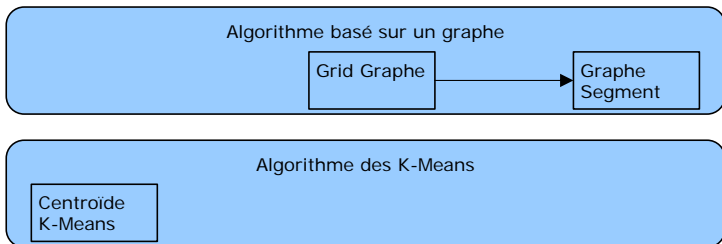
Algorithme basé sur un graphe

Grid Graphe

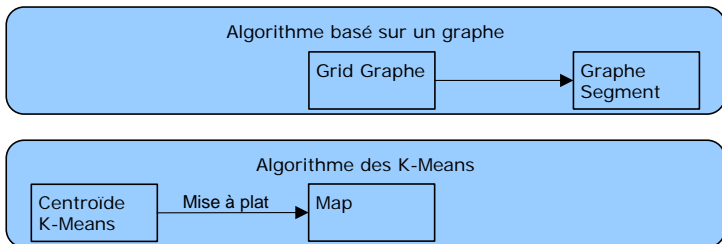
Workflow



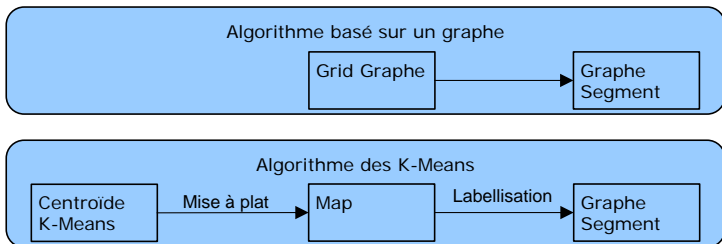
Workflow



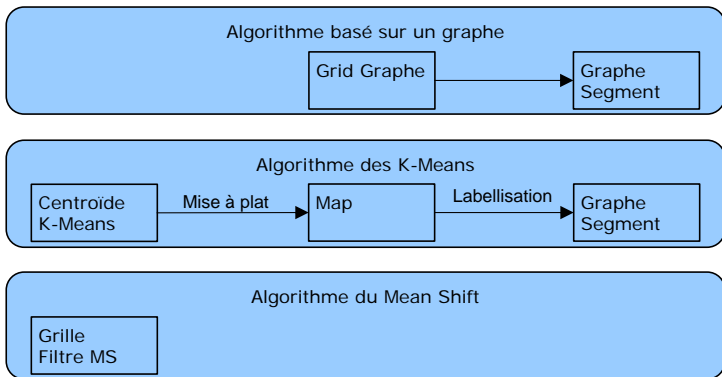
Workflow



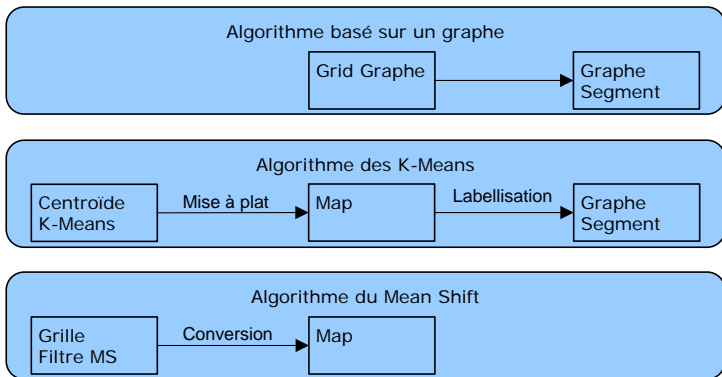
Workflow



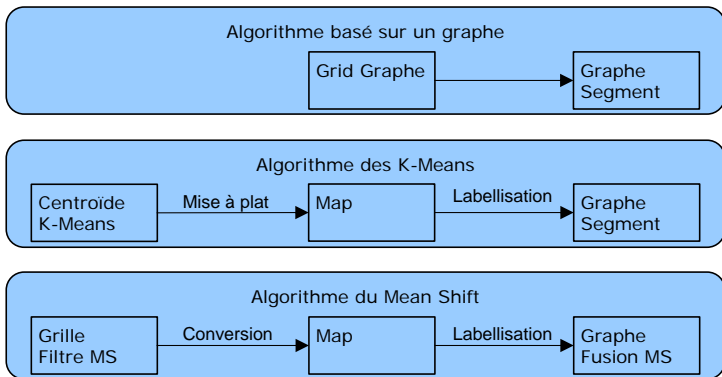
Workflow



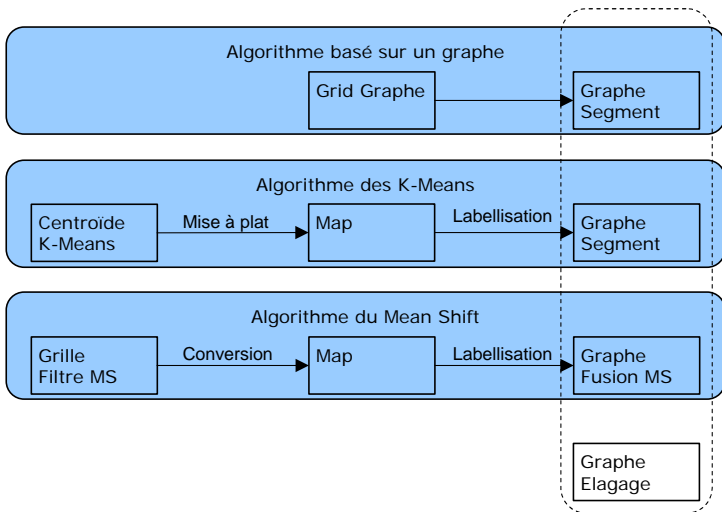
Workflow



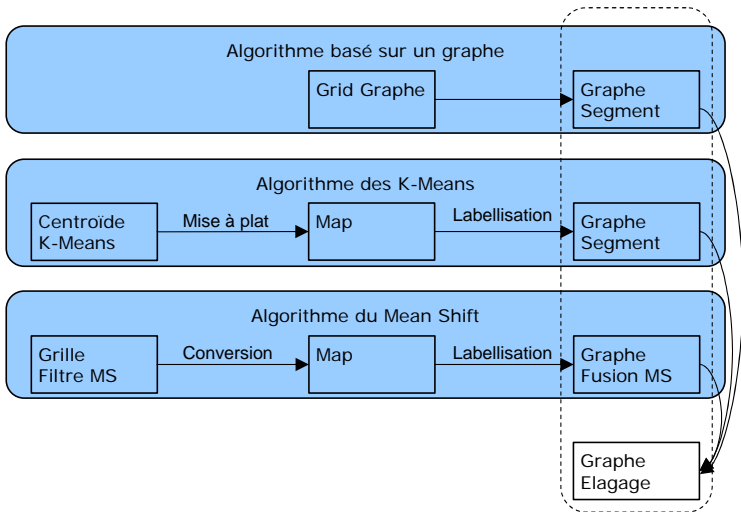
Workflow



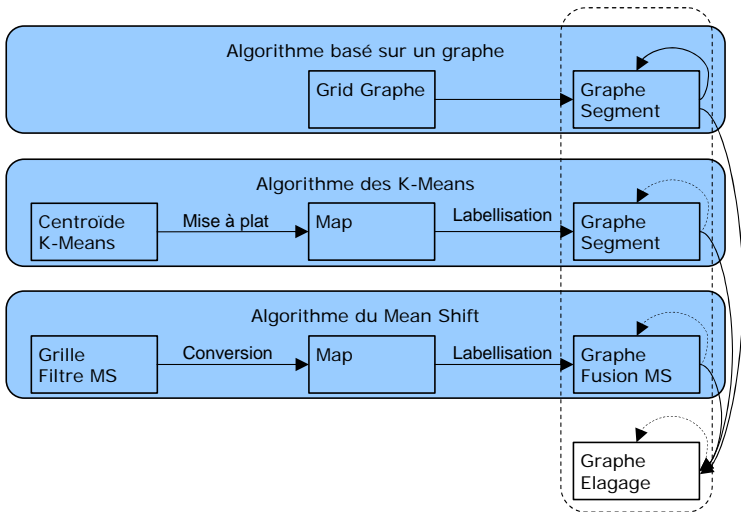
Workflow



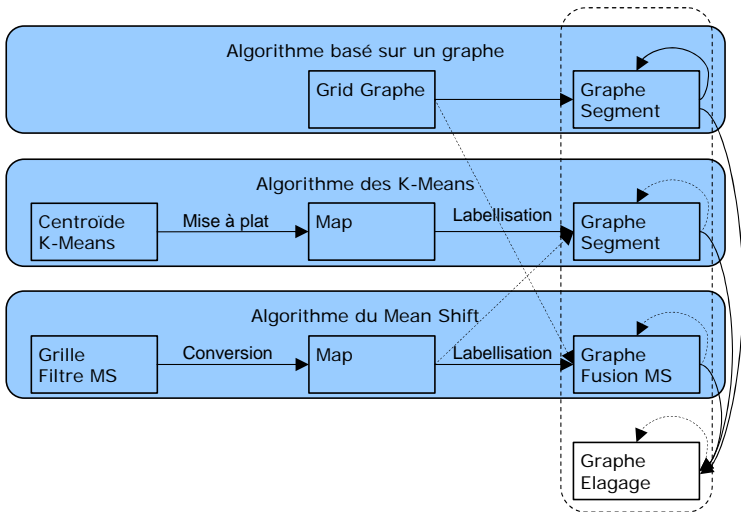
Workflow



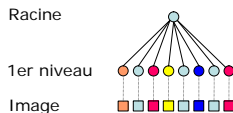
Workflow



Workflow



Pyramide de segmentation



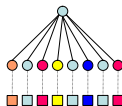
- On a besoin d'une structure de graphe.
- Cette structure doit aussi être hiérarchique.
- Implémentation efficaces sous formes d'ensemble disjoints dynamiques : réunion pondérée et compression de chemin.

Pyramide de segmentation

Racine

1er niveau

Image

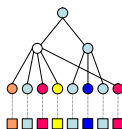


Racine

2ème niveau

1er niveau

Image



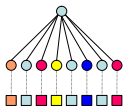
- On a besoin d'une structure de graphe.
- Cette structure doit aussi être hiérarchique.
- Implémentation efficaces sous formes d'ensemble disjoints dynamiques : réunion pondérée et compression de chemin.

Pyramide de segmentation

Racine

1er niveau

Image

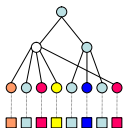


Racine

2ème niveau

1er niveau

Image



- On a besoin d'une structure de graphe.
- Cette structure doit aussi être hiérarchique.
- Implémentation efficaces sous formes d'ensemble disjoints dynamiques : réunion pondérée et compression de chemin.

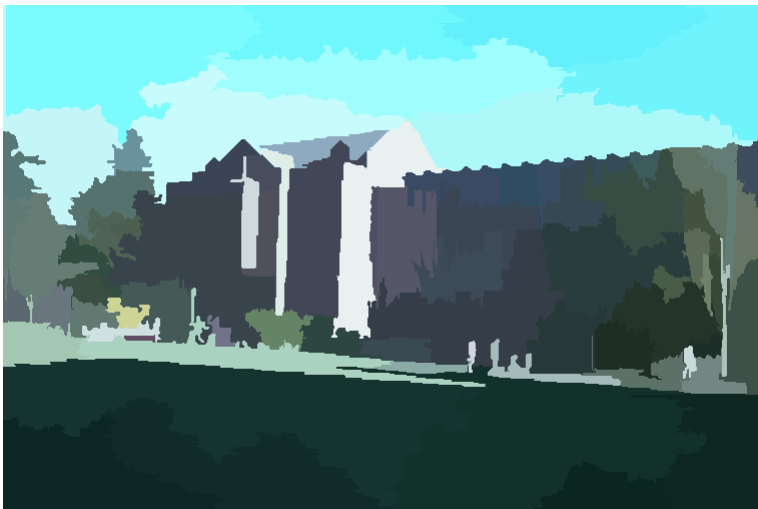
Segmentation pyramidale



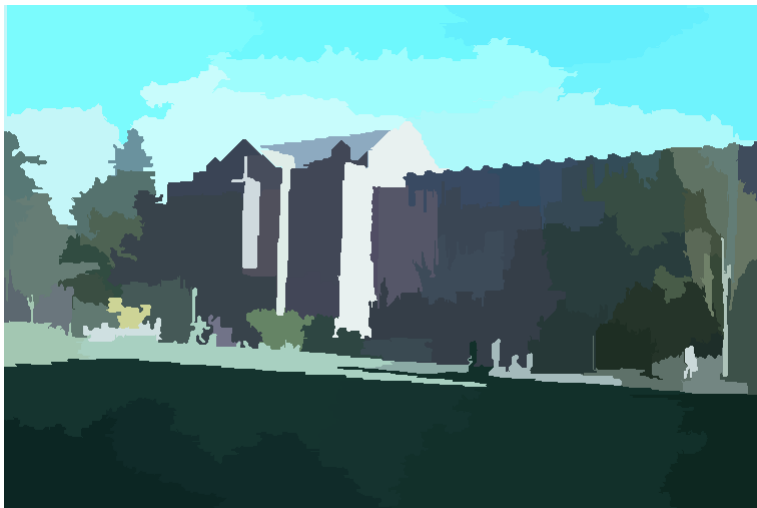
Segmentation pyramidale



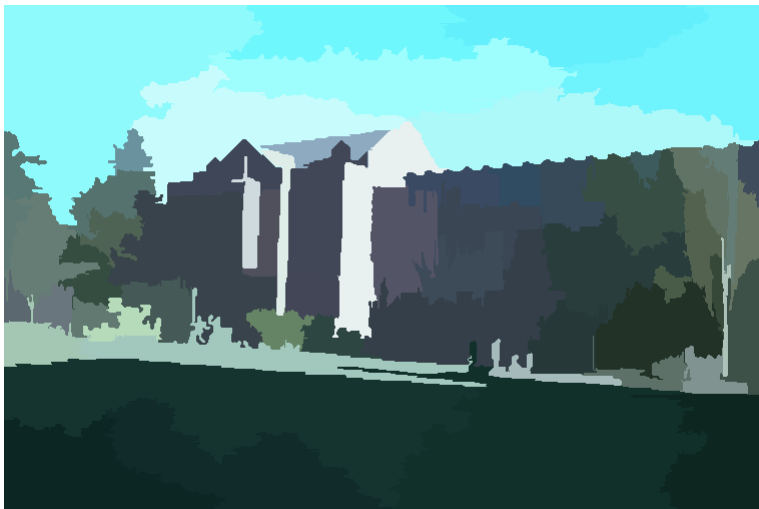
Segmentation pyramidale



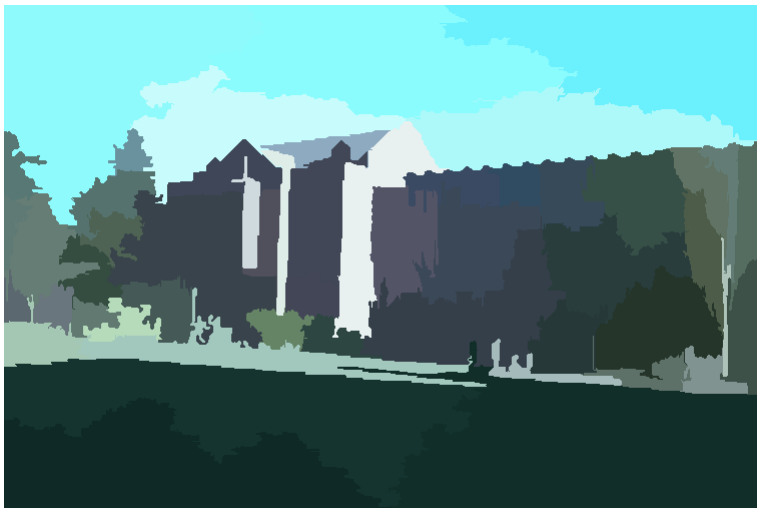
Segmentation pyramidale



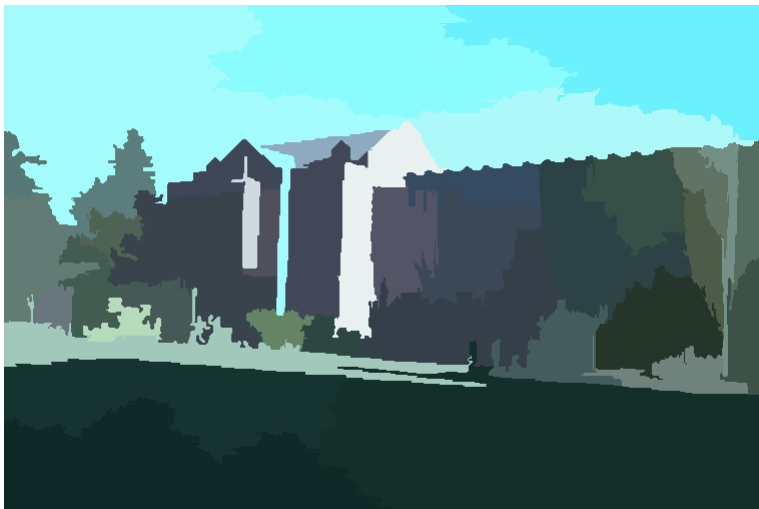
Segmentation pyramidale



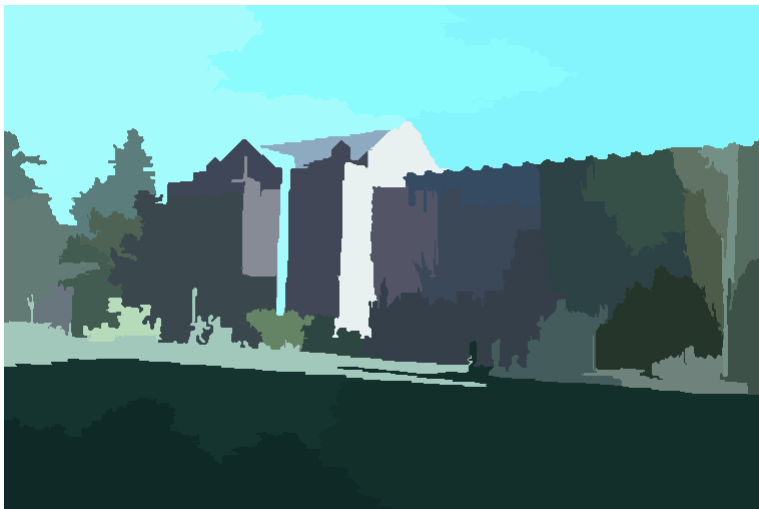
Segmentation pyramidale



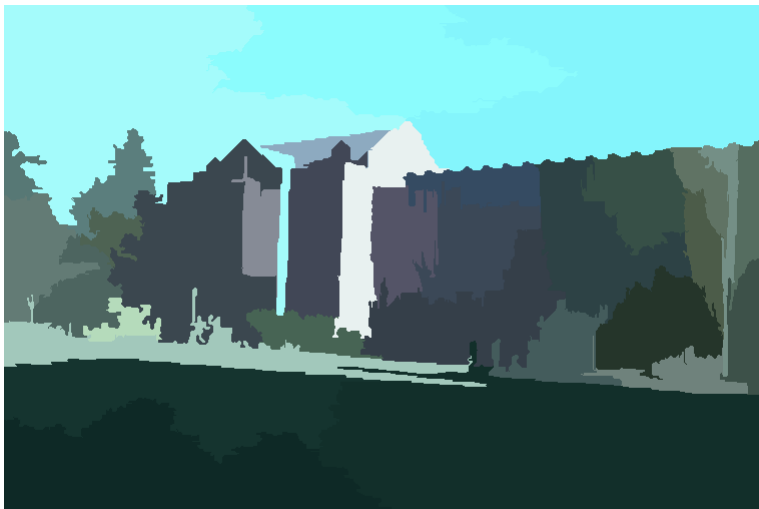
Segmentation pyramidale



Segmentation pyramidale

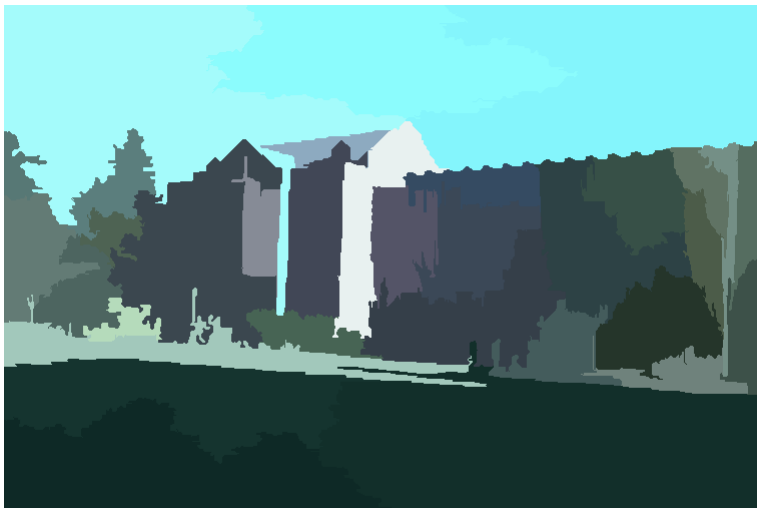


Segmentation pyramidale



Segmentation pyramidale

► Début



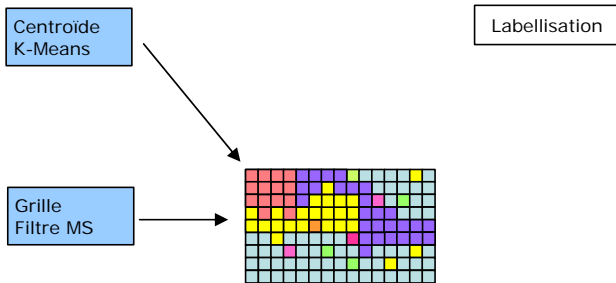
Labellisation et recherche des frontières

Centroïde
K-Means

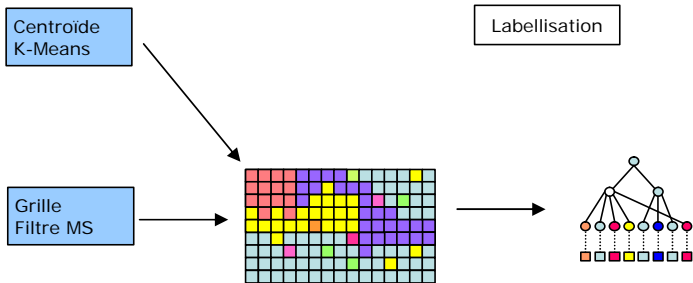
Labellisation

Grille
Filtre MS

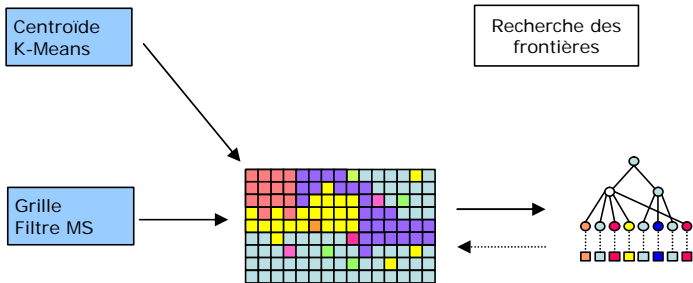
Labellisation et recherche de frontières



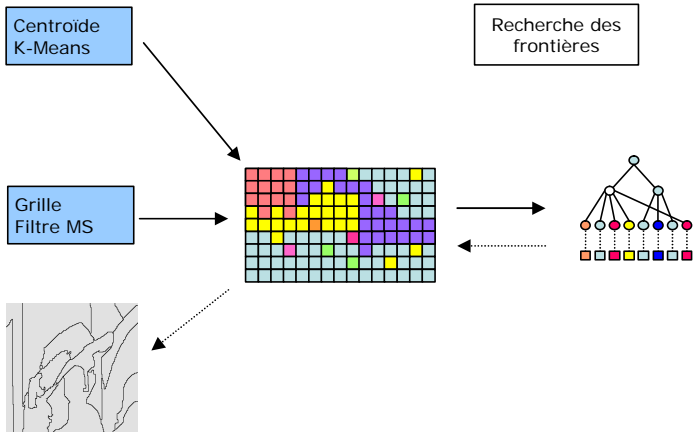
Labellisation et recherche de frontières



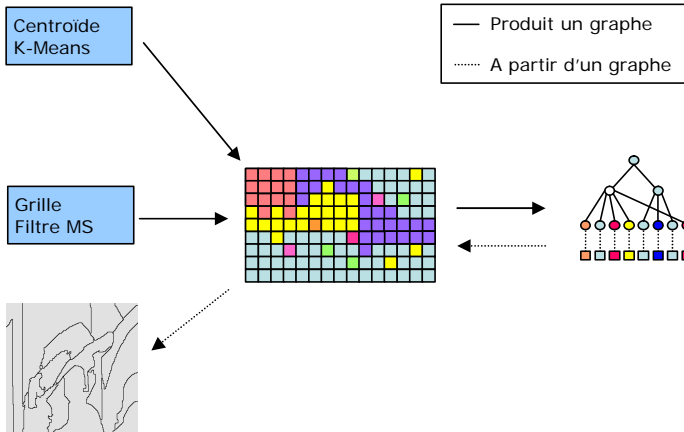
Labellisation et recherche de frontières



Labellisation et recherche de frontières



Labellisation et recherche de frontières



Résultat 1



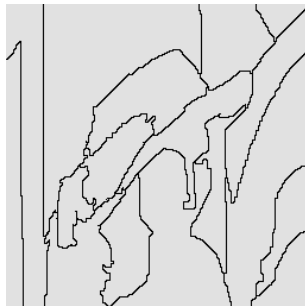
- 1 Palette de couleurs : "cool" de Matlab
- 2 Palette fixe : couleurs cycliques

Résultat 2



- 1 Couleurs calculés (moyenne)
- 2 Parfois surprenant

Résultat 3



- ① Mise à plat du graphe
- ② Parcours

Résultat 4



- 1 Graphe complet
- 2 Arête d'adjacence

Conclusions

- But globalement atteint :
 - programme en ligne de commande
 - robuste
 - documenté
- Il n'existe pas de programme équivalent
- Efficacité de la structure de graphe
- Outil à utiliser avec d'autres programmes : Matlab
- Possibilité d'évaluer le résultat sur un benchmark
- Les régions obtenues sont une base pour d'autres traitements

Conclusions

- But globalement atteint :
 - programme en ligne de commande
 - robuste
 - documenté
- Il n'existe pas de programme équivalent
- Efficacité de la structure de graphe
- Outil à utiliser avec d'autres programmes : Matlab
- Possibilité d'évaluer le résultat sur un benchmark
- Les régions obtenues sont une base pour d'autres traitements

Conclusions

- But globalement atteint :
 - programme en ligne de commande
 - robuste
 - documenté
- Il n'existe pas de programme équivalent
- Efficacité de la structure de graphe
- Outil à utiliser avec d'autres programmes : Matlab
- Possibilité d'évaluer le résultat sur un benchmark
- Les régions obtenues sont une base pour d'autres traitements

Conclusions

- But globalement atteint :
 - programme en ligne de commande
 - robuste
 - documenté
- Il n'existe pas de programme équivalent
- Efficacité de la structure de graphe
- Outil à utiliser avec d'autres programmes : Matlab
- Possibilité d'évaluer le résultat sur un benchmark
- Les régions obtenues sont une base pour d'autres traitements

Conclusions

- But globalement atteint :
 - programme en ligne de commande
 - robuste
 - documenté
- Il n'existe pas de programme équivalent
- Efficacité de la structure de graphe
- Outil à utiliser avec d'autres programmes : Matlab
- Possibilité d'évaluer le résultat sur un benchmark
- Les régions obtenues sont une base pour d'autres traitements

Conclusions

- But globalement atteint :
 - programme en ligne de commande
 - robuste
 - documenté
- Il n'existe pas de programme équivalent
- Efficacité de la structure de graphe
- Outil à utiliser avec d'autres programmes : Matlab
- Possibilité d'évaluer le résultat sur un benchmark
- Les régions obtenues sont une base pour d'autres traitements