

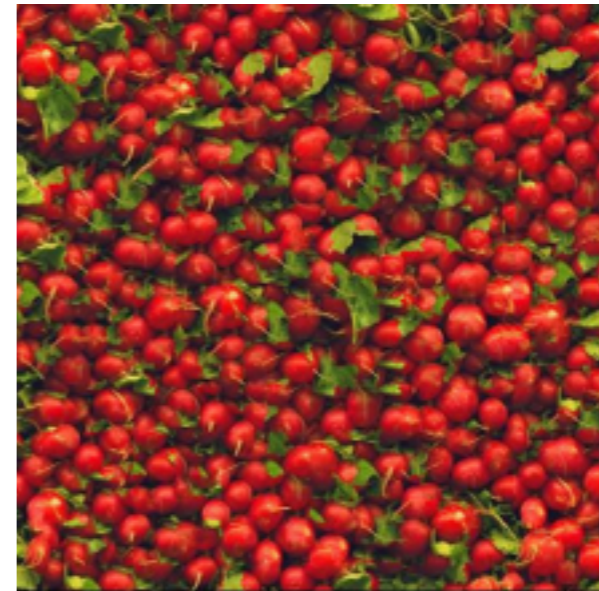
# Découpage d'images



GIF-4105/7105 Photographie Algorithmique, Hiver 2015  
Jean-François Lalonde

# La semaine dernière

- Synthèse de textures

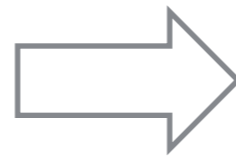
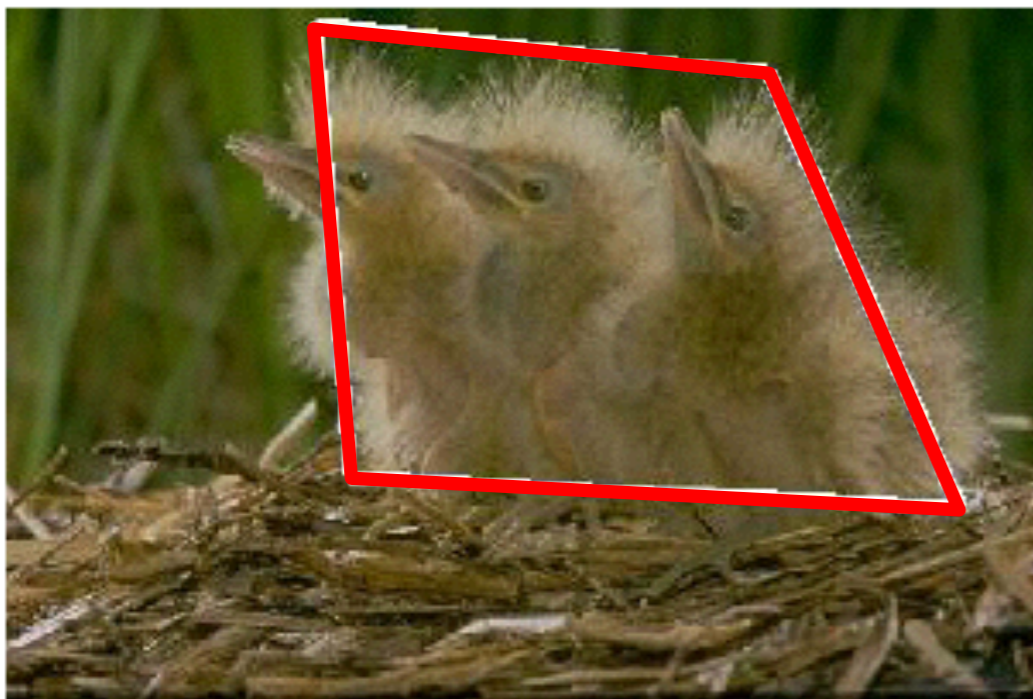


- Re-dimensionnement d'images



# Aujourd'hui

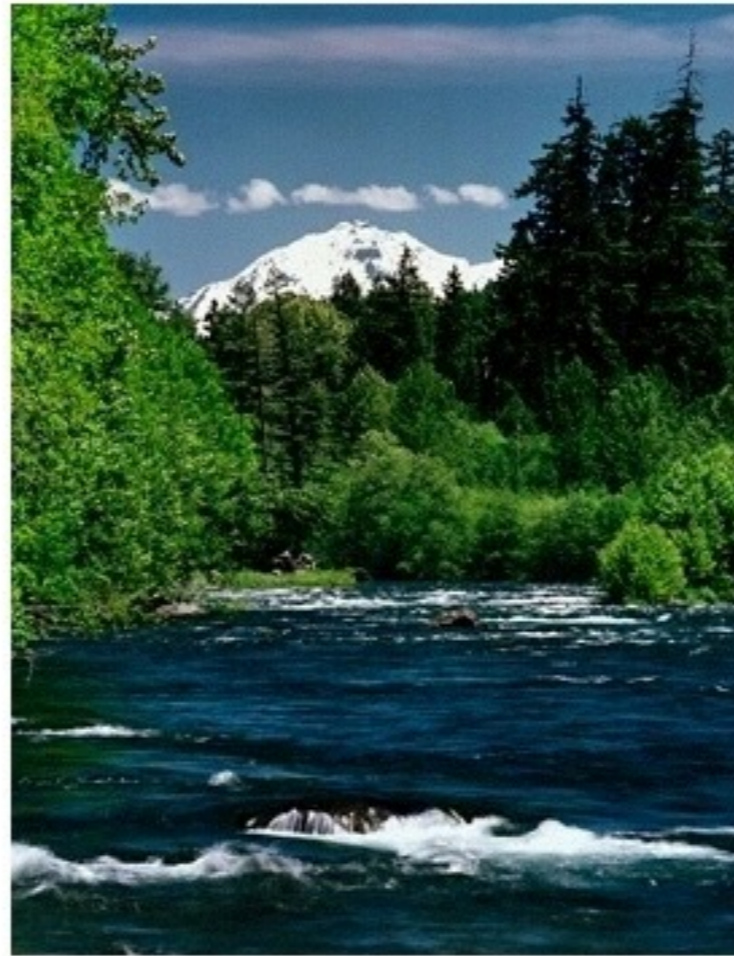
Segmentation (ou découpage) d'images



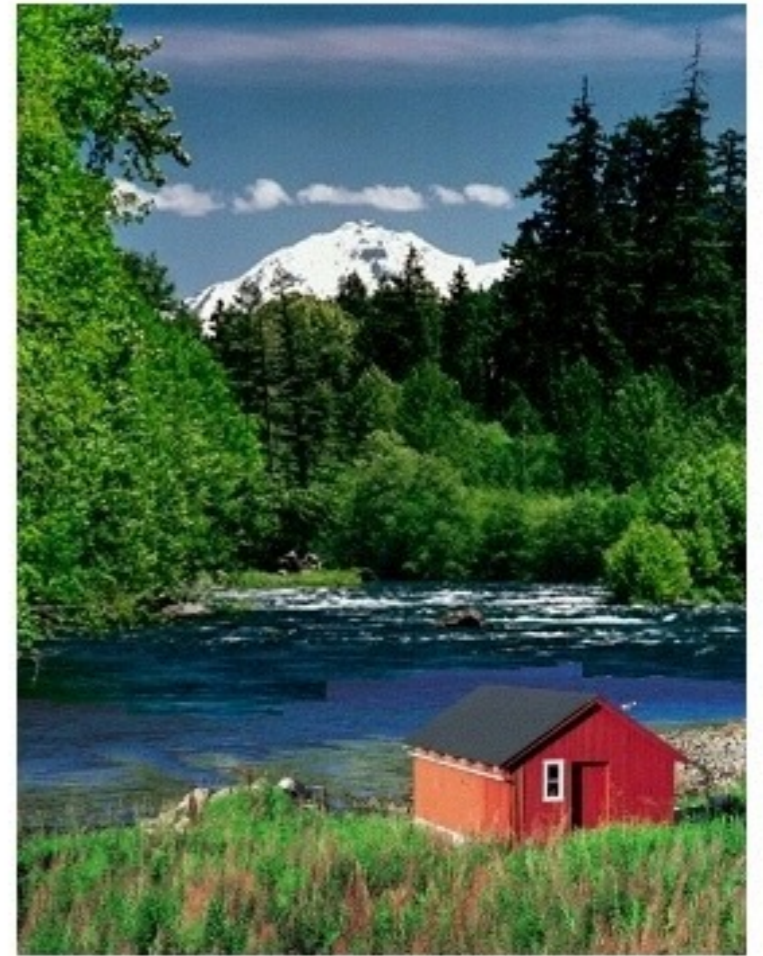
# Application: composer les images



+

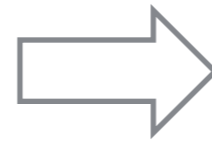
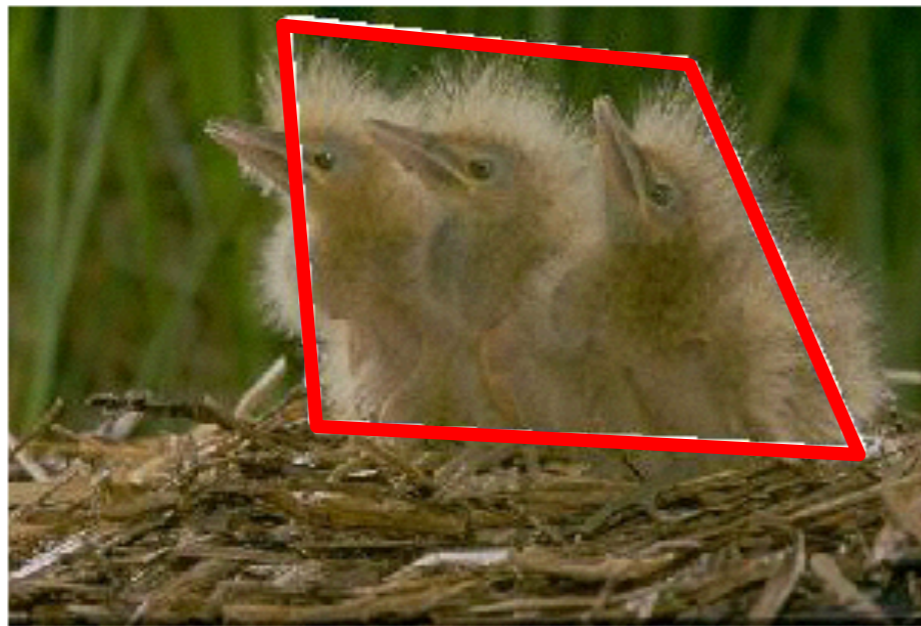


=



# Segmentation semi-automatique

- L'utilisateur n'indique ses préférences qu'approximativement — l'algorithme doit faire le reste!



- Problèmes:
  - Quels groupes de pixels forment des régions cohérentes?
  - Quels pixels sont probablement à la frontière de ces régions?
  - Quelle région l'utilisateur désire-t-il?

# Les caractéristiques d'une "bonne" région?

- Contient couleurs et textures similaires
- Différente de l'arrière-plan
- Compacte

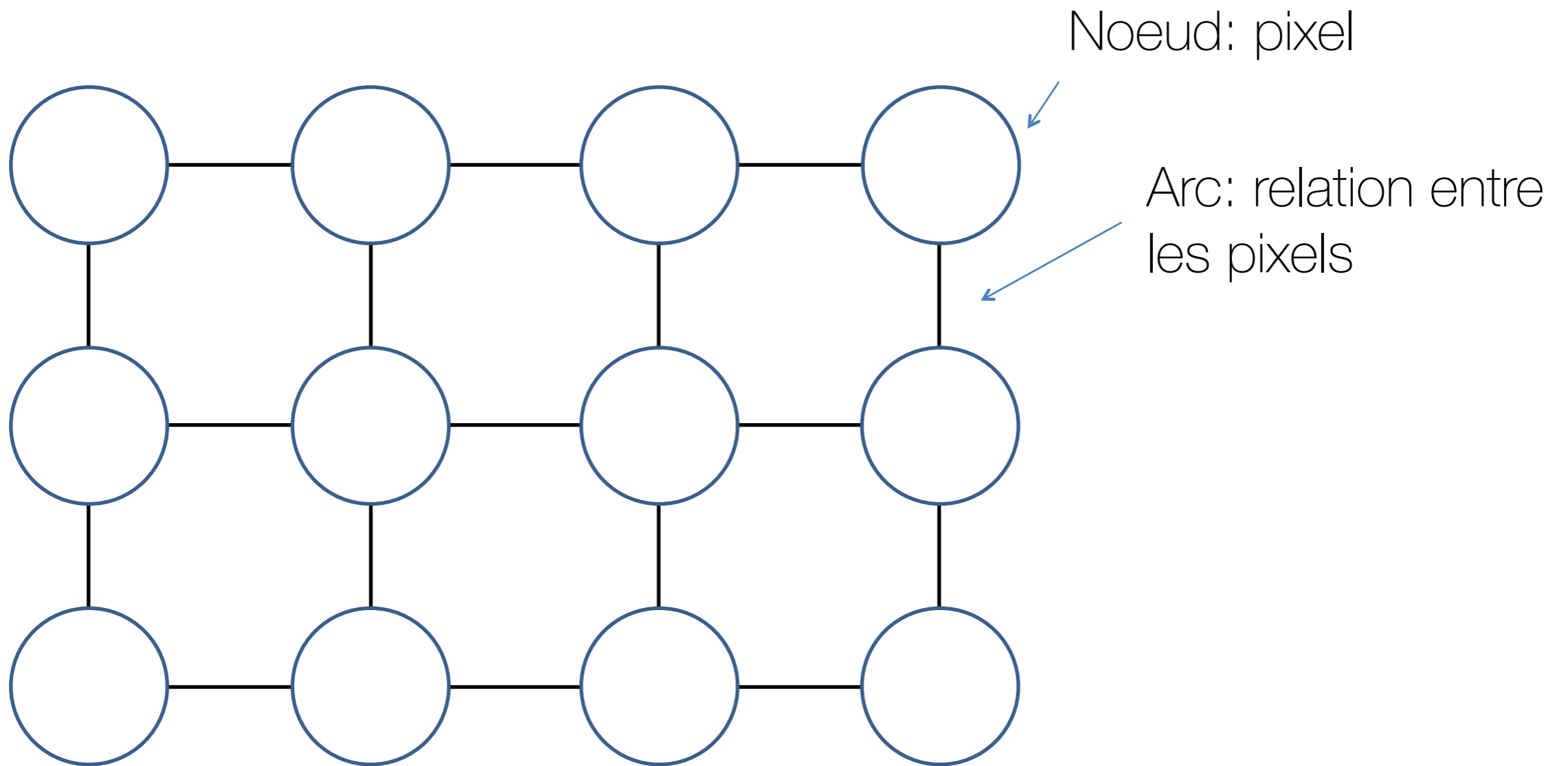


# Les caractéristiques d'une "bonne" frontière?

- Gradient élevé
- Gradient dans la bonne direction
- Lisse et continue



# Idée principale: l'image est un graphe!





# L'image est un graphe!

## **Synthèse de textures**

Noeuds: différence de chevauchement

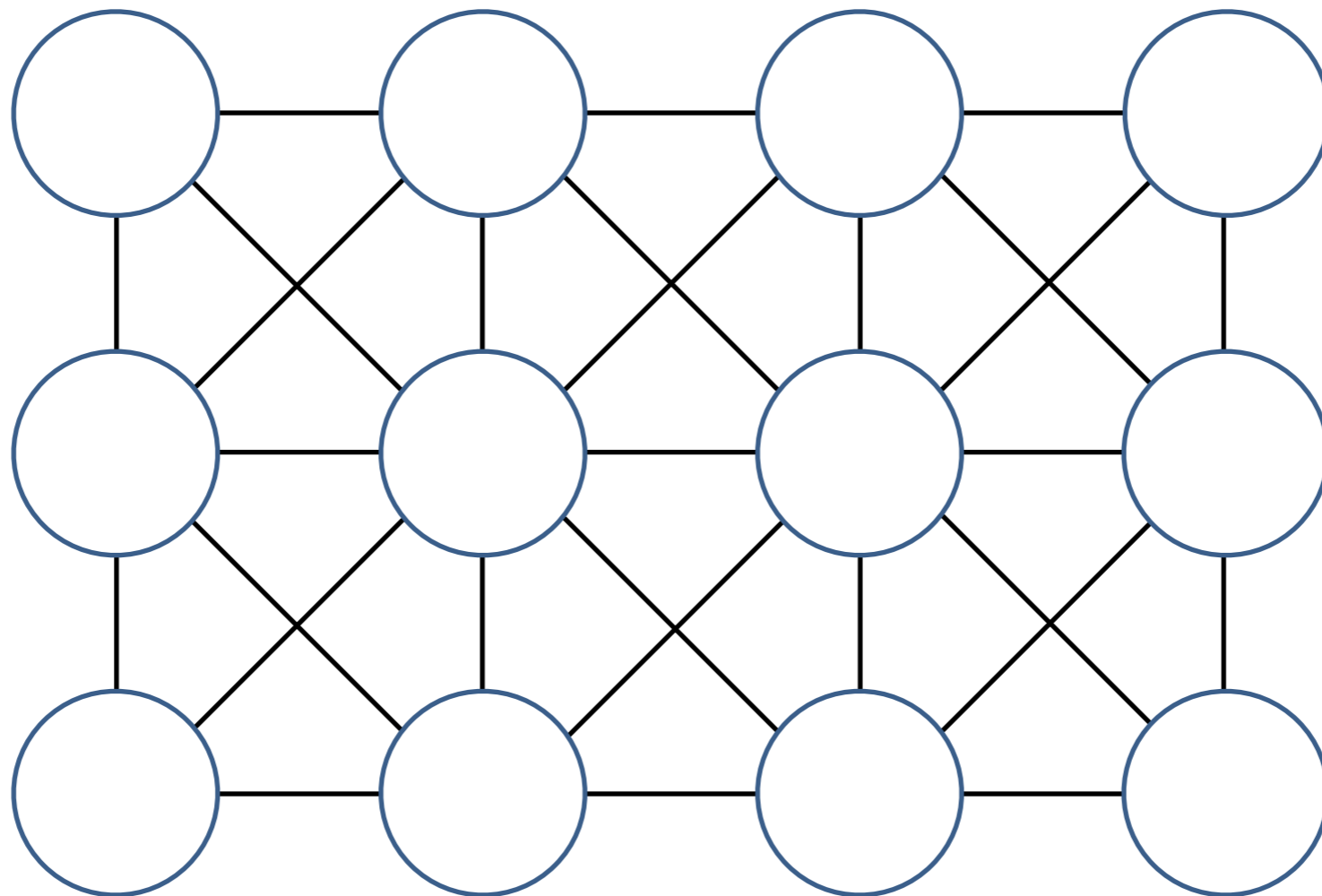
Arcs: voisinage

## **Redimensionnement**

Noeuds: "importance" d'un pixel

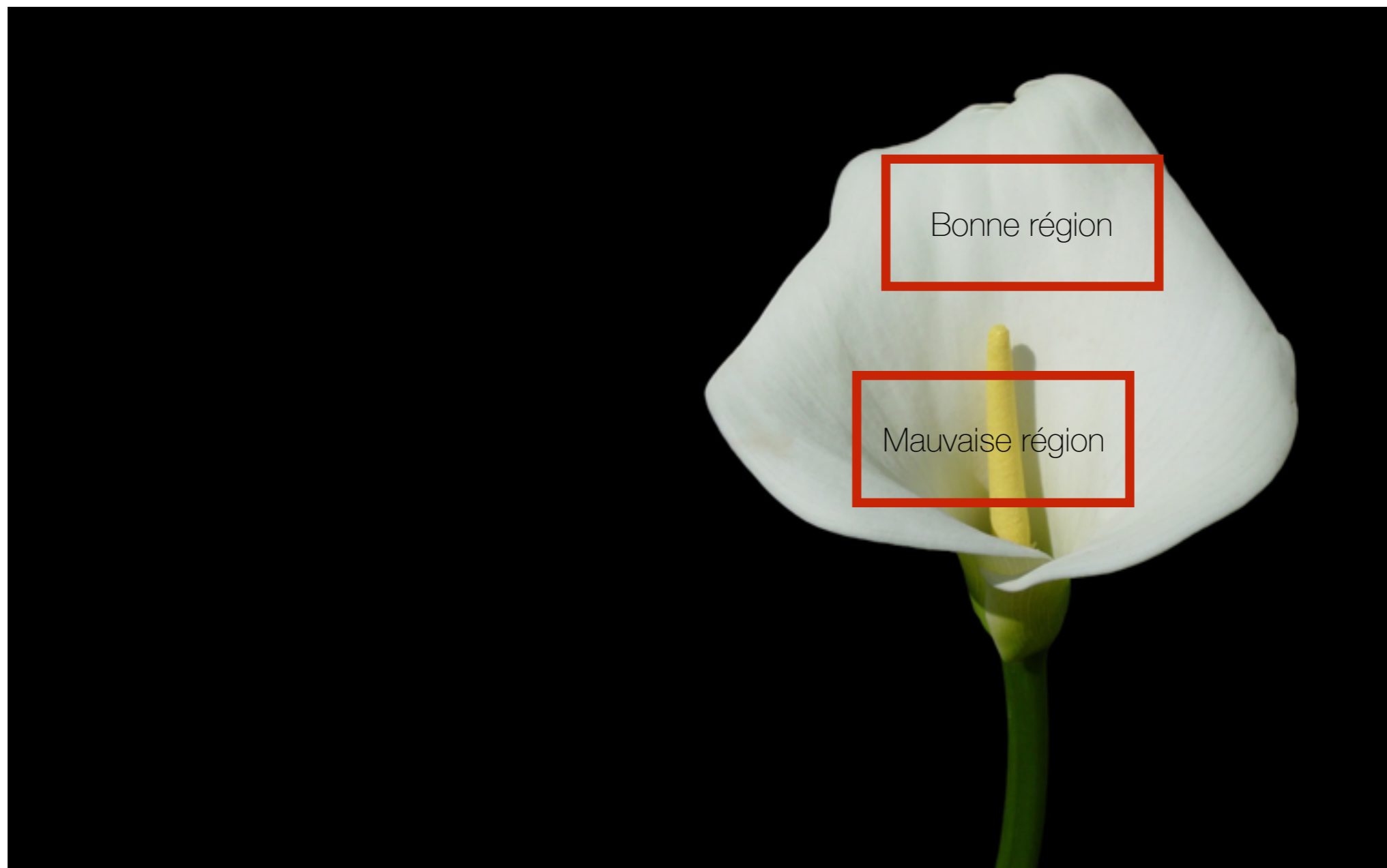
Arcs: voisinage

Algorithme: trouver chemin au coût total minimum



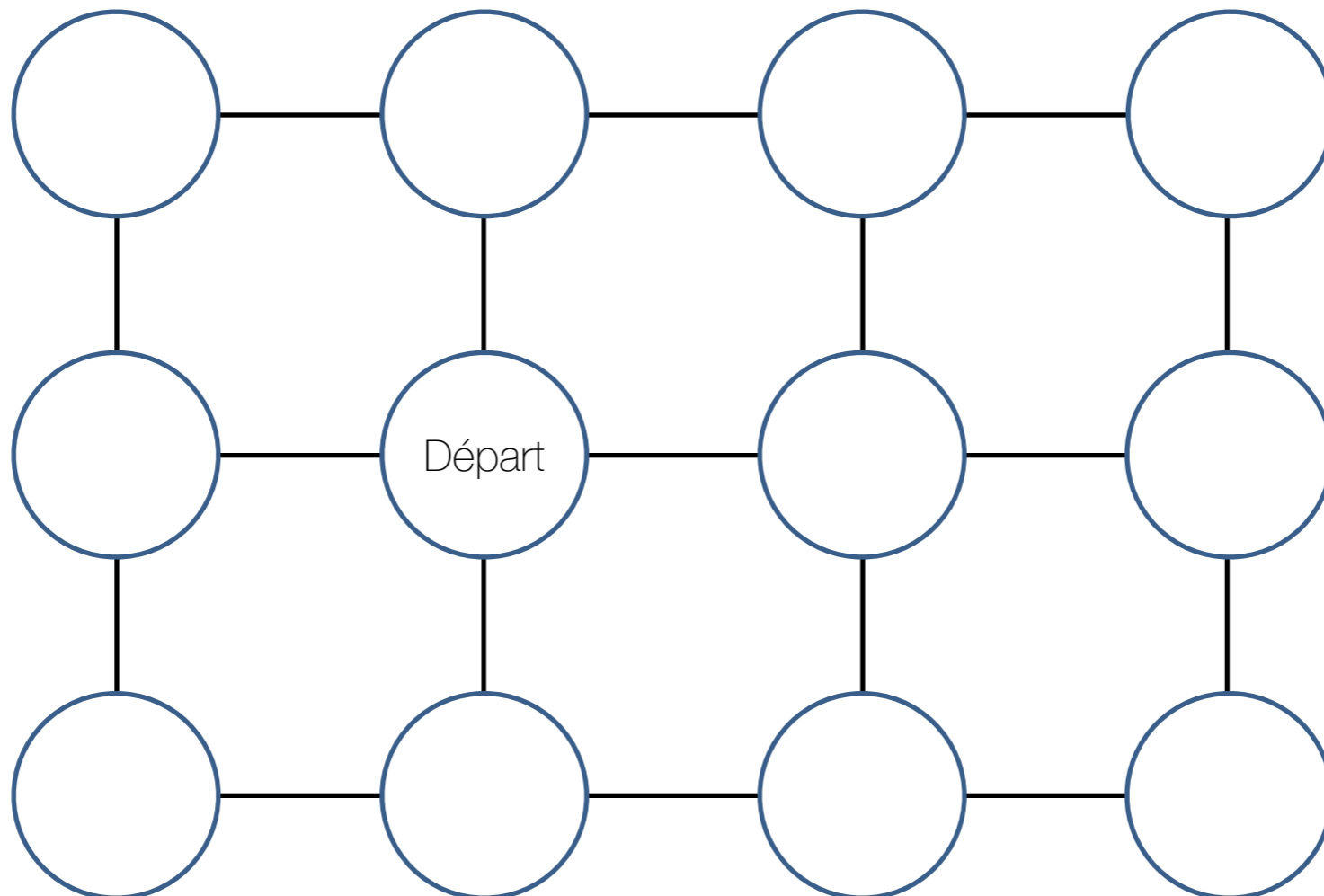
# Modéliser les régions: Baguette magique

- Modèle: couleur doit être similaire



# Baguette magique

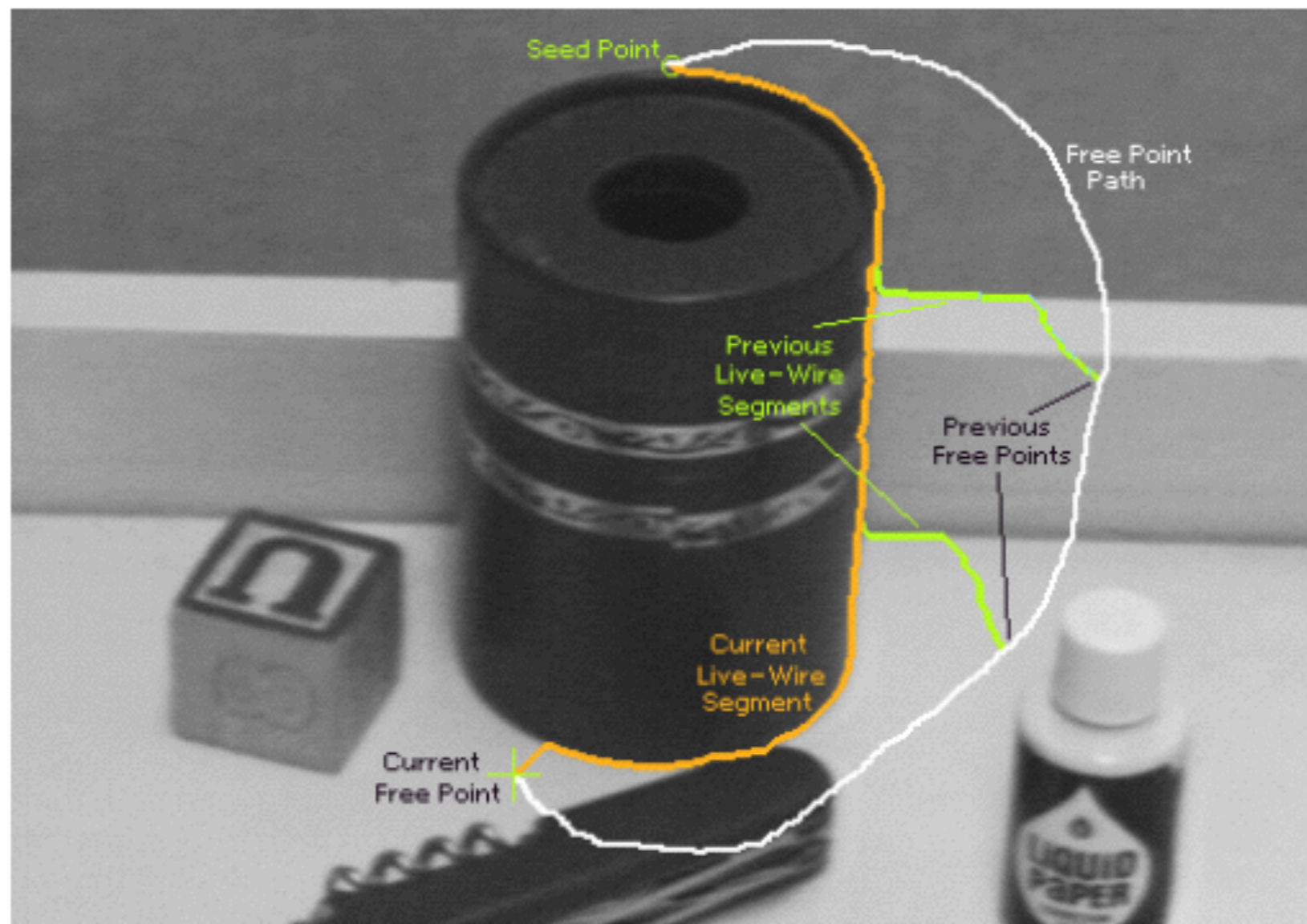
- Algorithme:
  - Cliquer sur un pixel
  - Tant que la couleur est similaire au pixel original
    - Étendre la sélection en parcourant le graphe vers l'extérieur



Démonstration (gimp)

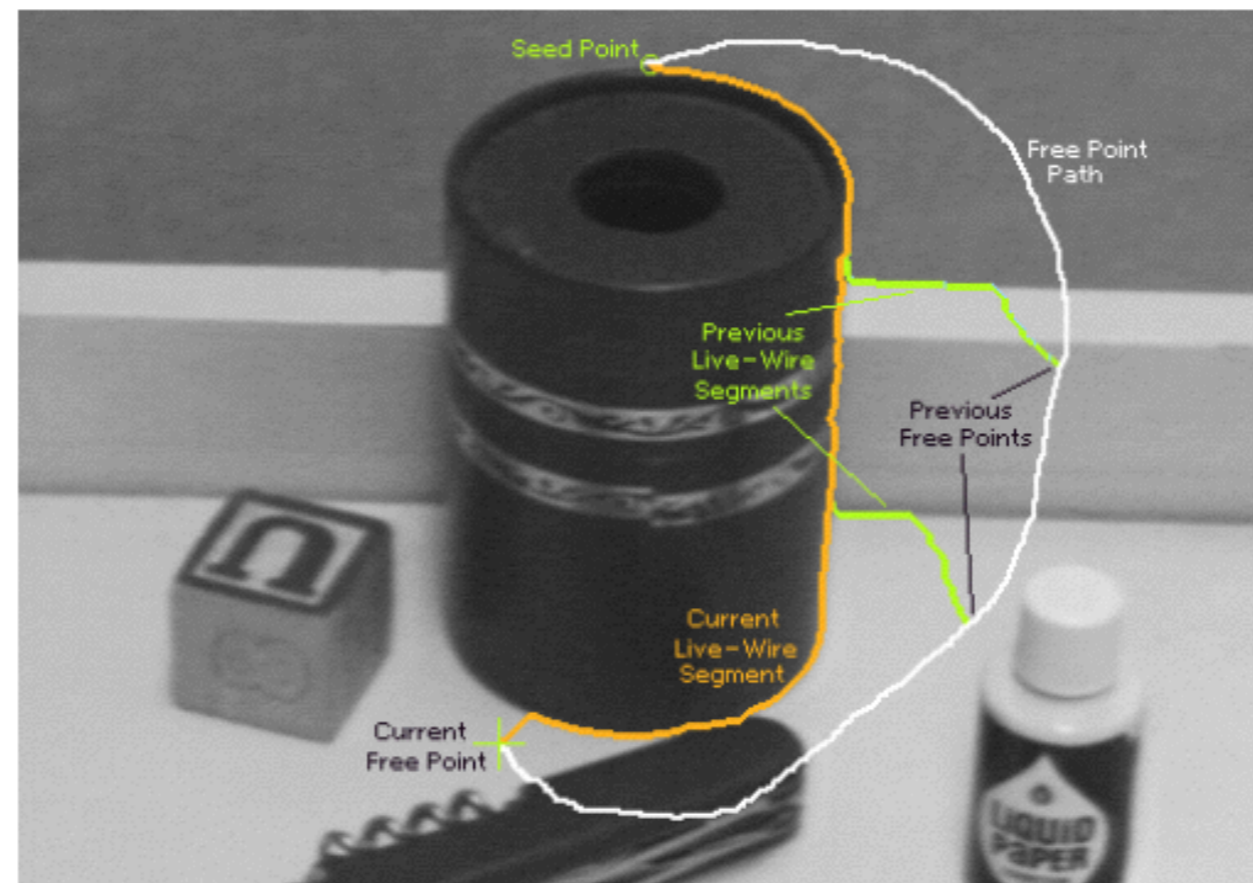
# Ciseaux intelligents

Mortenson and Barrett, Intelligent Scissors, SIGGRAPH 1995



# Ciseaux intelligents

- But: trouver une bonne frontière entre deux points
- Défis
  - Définir: qu'est-ce qu'une bonne frontière?
  - Calculer le chemin rapidement pour minimiser le temps d'interaction



# Qu'est-ce qu'une bonne frontière?

- Considérons deux pixels adjacents dans l'image. Il est probable qu'il y ait une frontière entre les deux si:
  - il y a une arête
    - `edge(im, 'canny')`
  - le gradient est fort
  - le gradient est dans la direction de la frontière

# Gradients, arêtes, et coûts

Magnitude des gradients



Coût (faible = frontière)



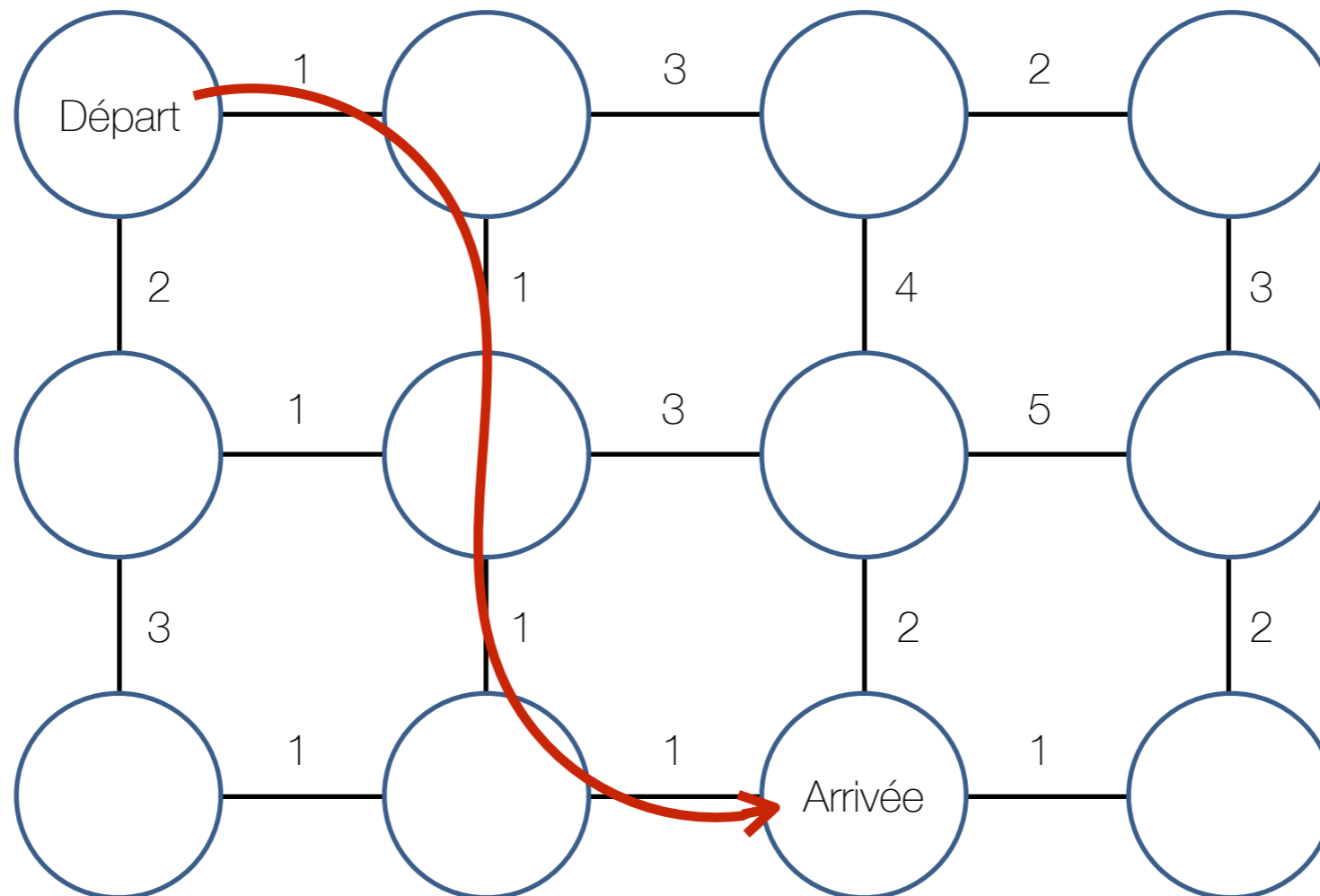
Arêtes





# Ciseaux intelligents

- Calculer le “coût de frontière” pour chaque pixel
- Trouver le chemin minimisant ce coût entre le point de départ et point courant



# Algorithme: ciseaux intelligents

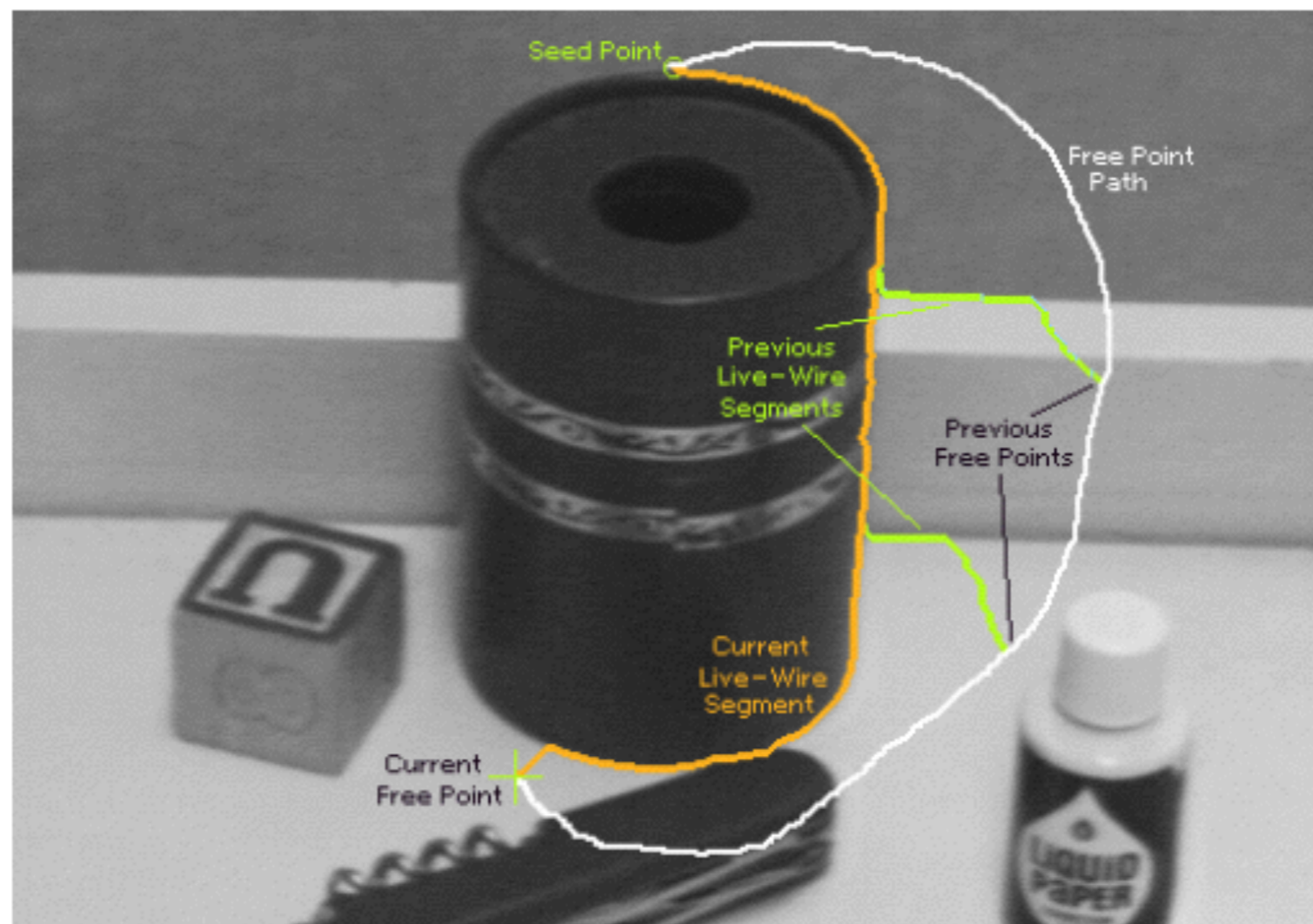
- Calculer le “coût de frontière” pour chaque pixel;
- L'utilisateur clique sur un point de départ (origine);
- Tant que l'utilisateur déplace le curseur:
  - Calculer le plus court chemin entre l'origine et le curseur;
    - Algorithme de Dijkstra

# L'algorithme de Dijkstra

- Initialisation, étant donné origine  $o$ 
  - $\text{cout}(o) = 0$ ,  $\text{cout}(\text{autres pixels}) = \text{Inf}$
  - $A = \{o\}$ , pixels à visiter
  - $V = \{ \}$ , pixels visités
  - $P(q)$ , pointeur vers pixel qui mène vers  $q$
- Tant que  $A$  n'est pas vide:
  - $q$  = pixel dans  $A$  au coût minimum
  - Retirer  $q$  de  $A$ , rajouter dans  $V$  (nous l'avons visité)
  - Pour chaque pixel  $p$  dans le voisinage de  $q$  et qui n'est pas dans  $V$ :
    - $\text{cout\_tmp} = \text{cout}(q) + \text{cout2}(q,p)$
    - Si ( $r$  n'est pas dans  $A$ ) OU ( $\text{cout\_tmp} < \text{cout}(p)$ )
      - $\text{cout}(p) = \text{cout\_tmp}$
      - $P(p) = q$
      - Rajouter  $p$  à  $A$

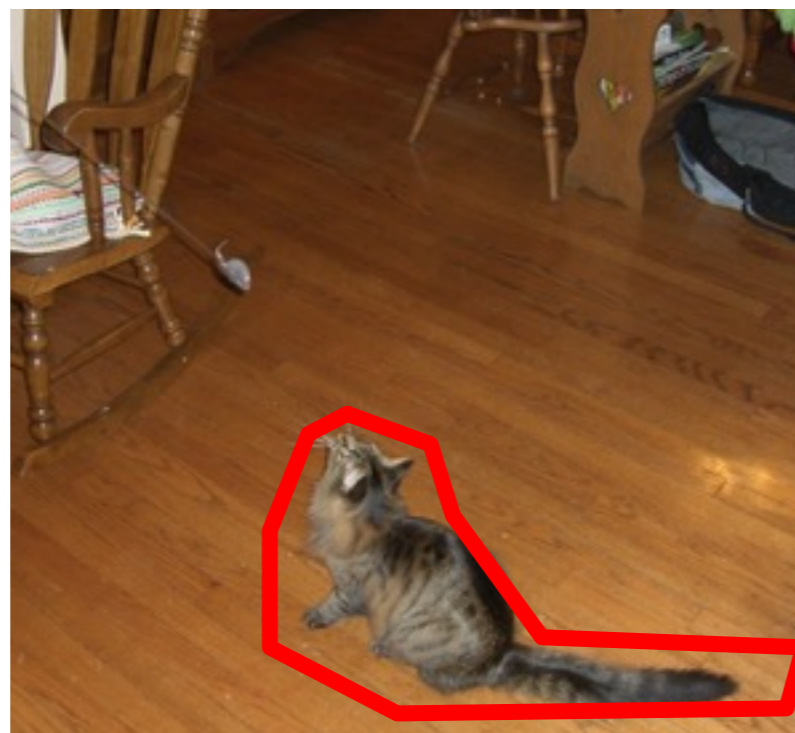
# Ciseaux intelligents: améliorations

- Premier départ: “sauter” vers la frontière la plus proche
- Ajuster la frontière de façon interactive quand l'utilisateur bouge le curseur
- Geler le chemin et créer de nouveaux points d'origine

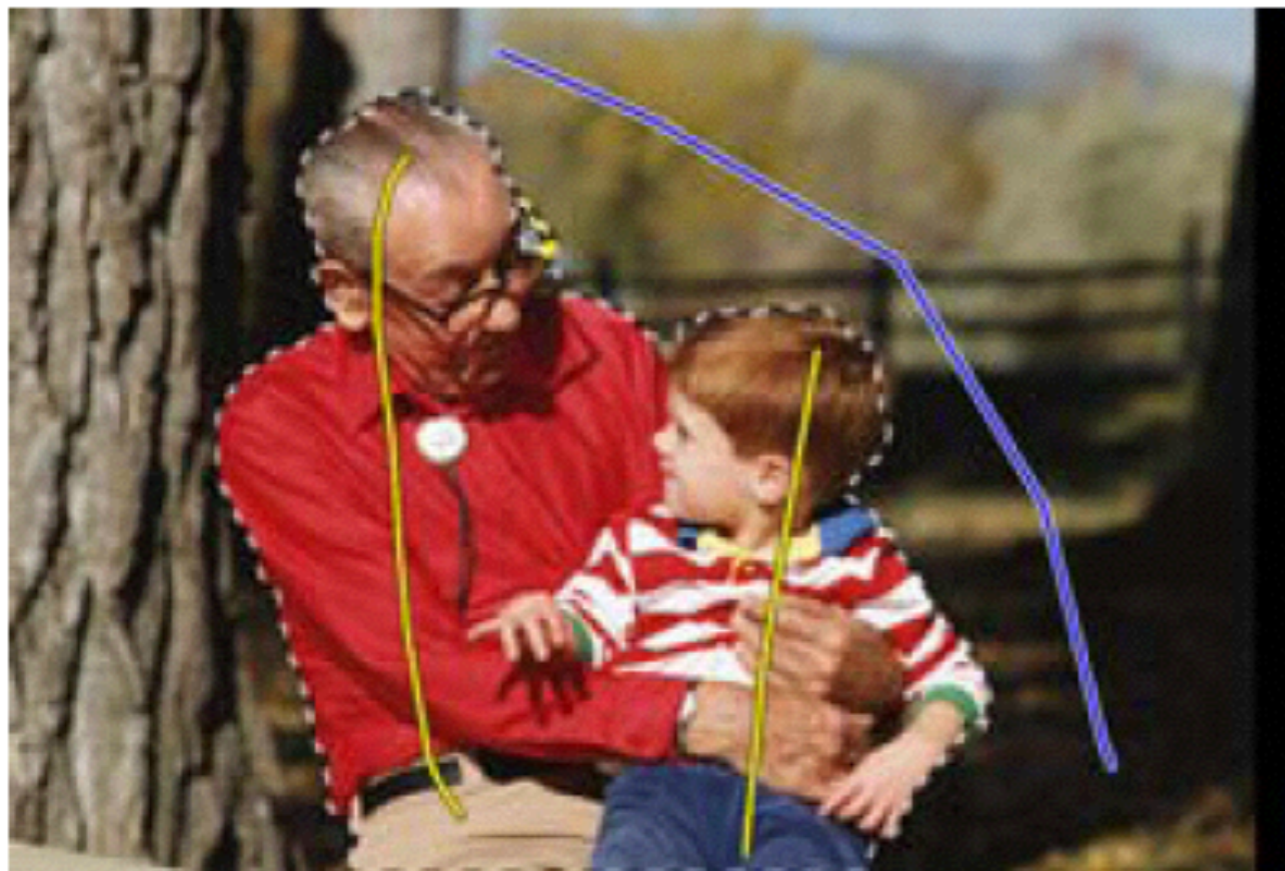


Démonstration (gimp)

# Quand est-ce que ça marche?



# Segmentation par coupure de graphe

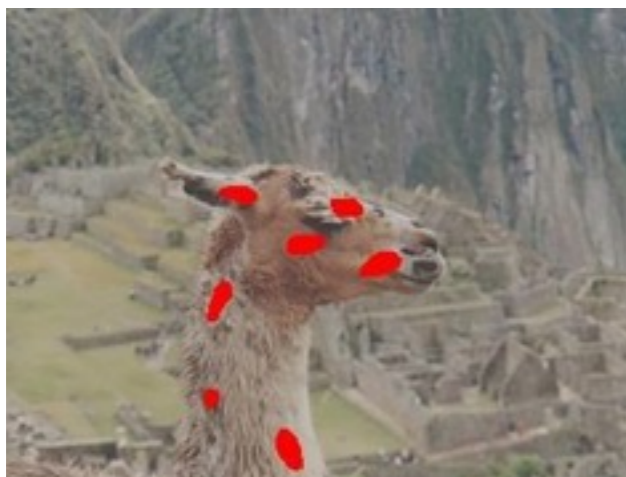


Rother et al., "Grab Cut", SIGGRAPH 2004  
Li et al., "Lazy Snapping", SIGGRAPH 2004

# Segmentation par coupure de graphe

- Idée: combiner régions & frontières

Baguette magique (198?)



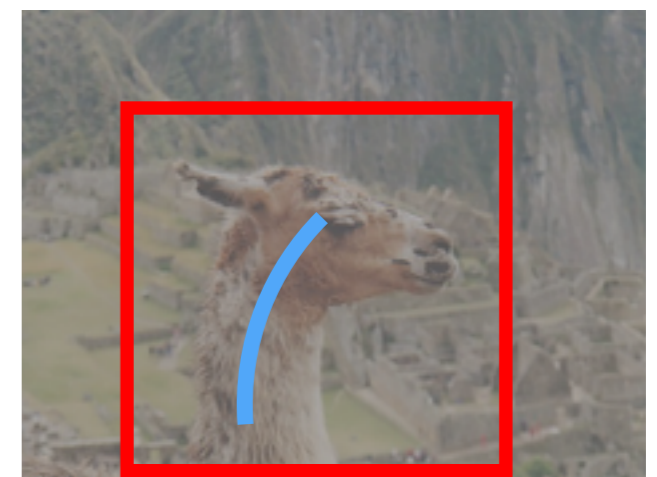
Régions

Ciseaux intelligents (1995)



Frontières

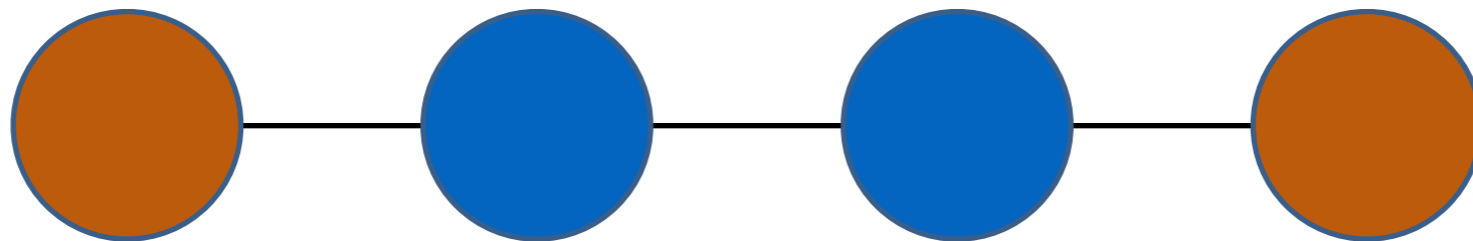
Segmentation interactive (2004)



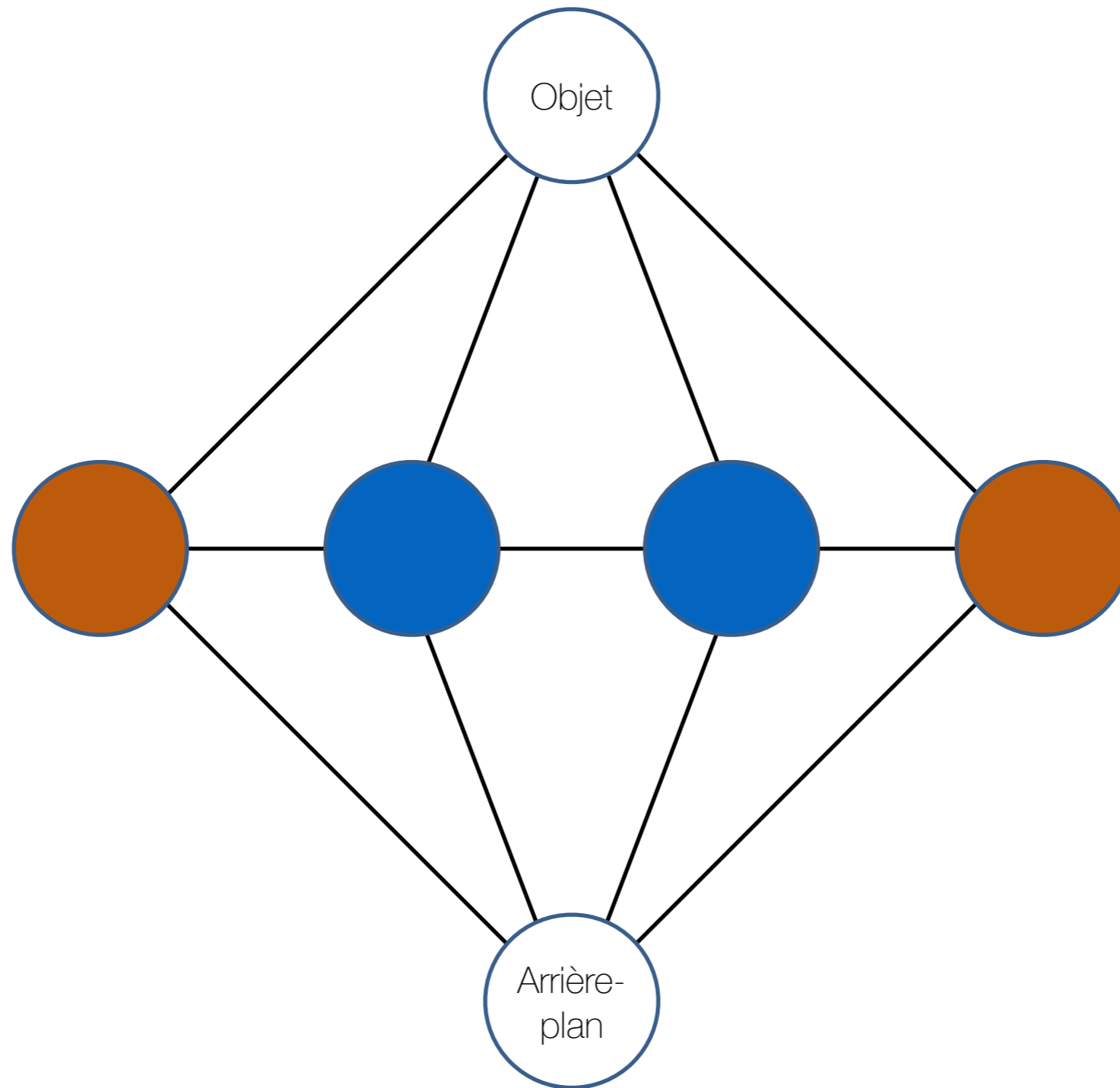
Régions & frontières



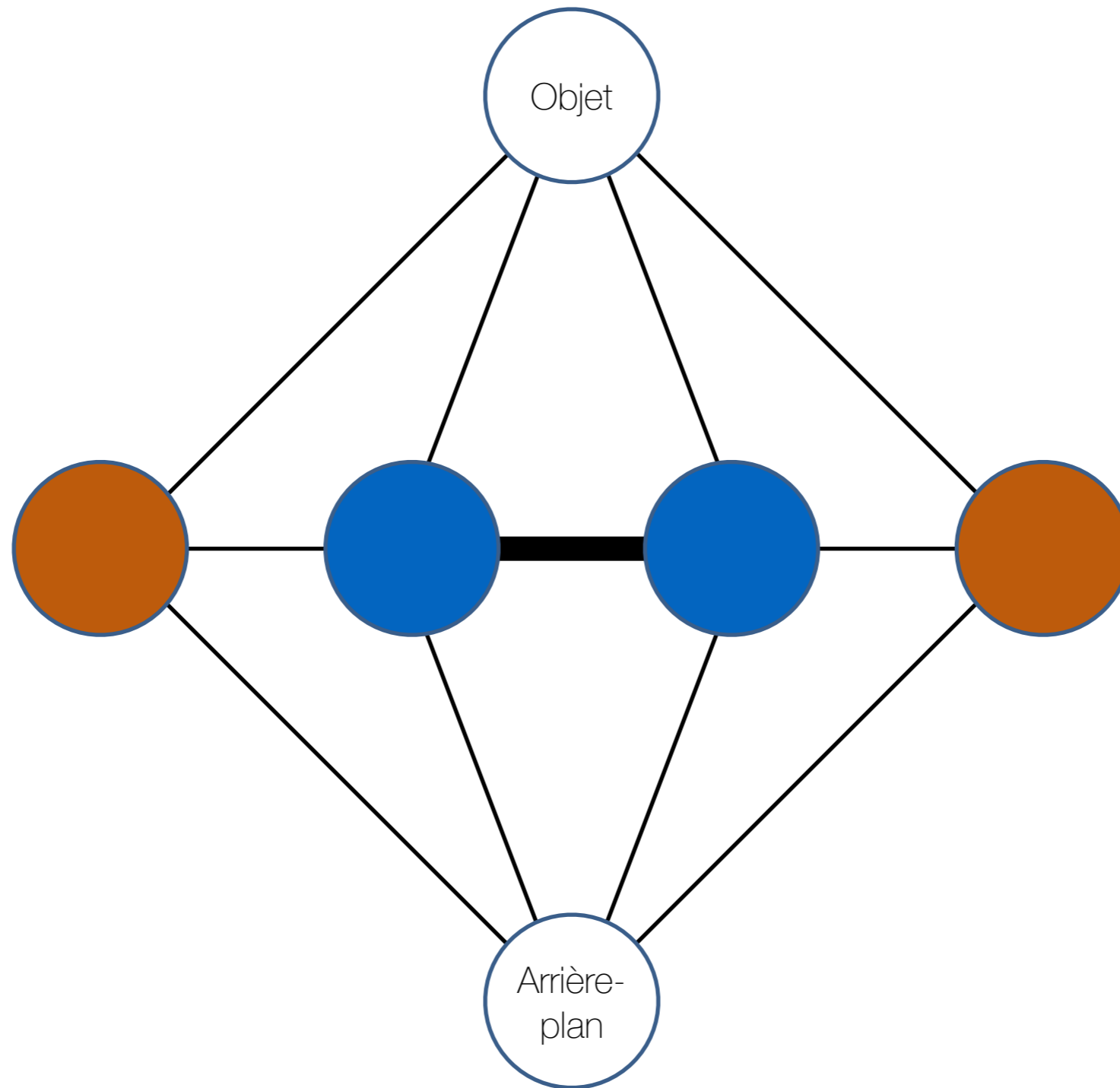
# Segmentation par coupure de graphe



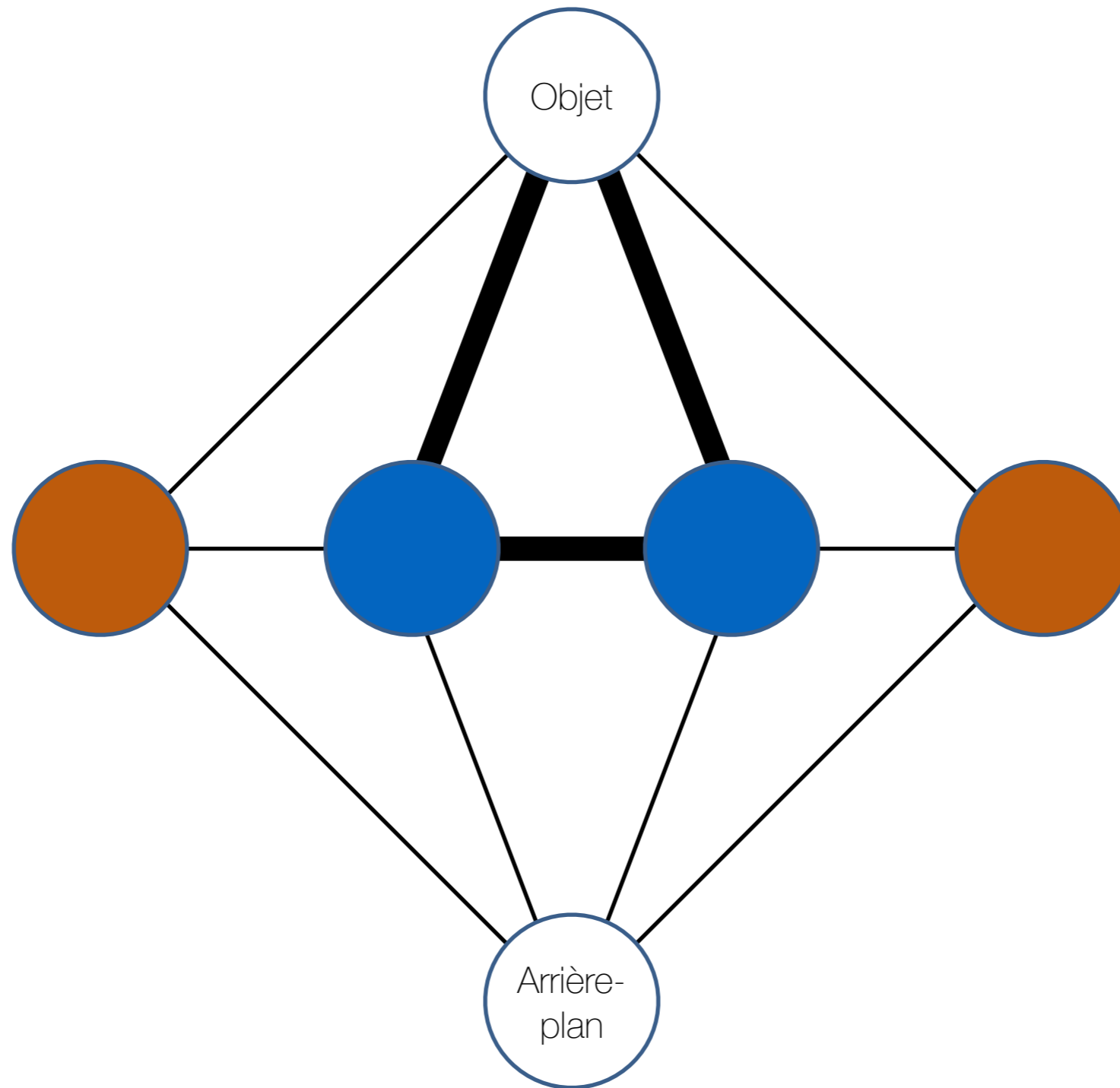
# Segmentation par coupure de graphe



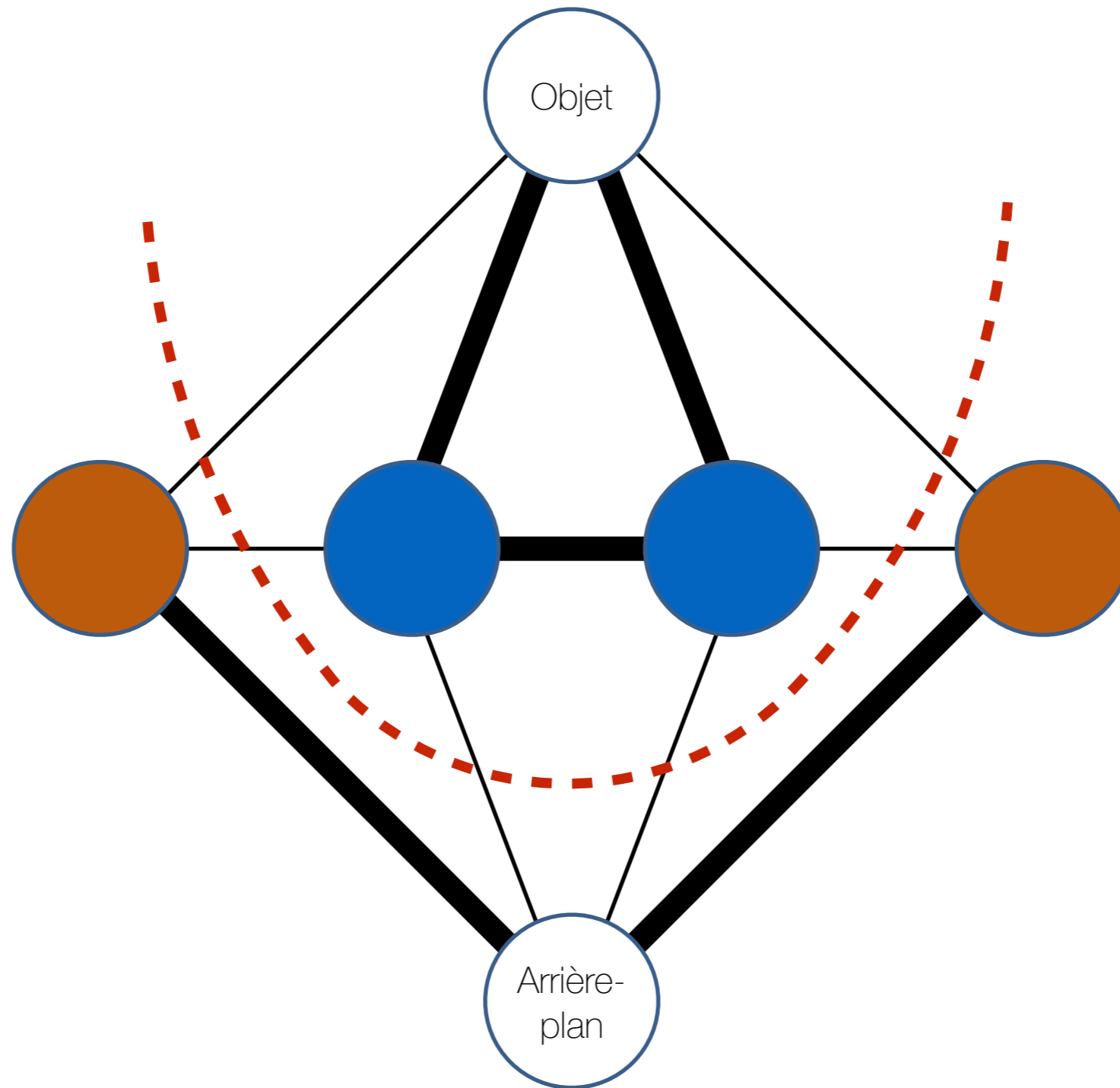
# Segmentation par coupure de graphe



# Segmentation par coupure de graphe

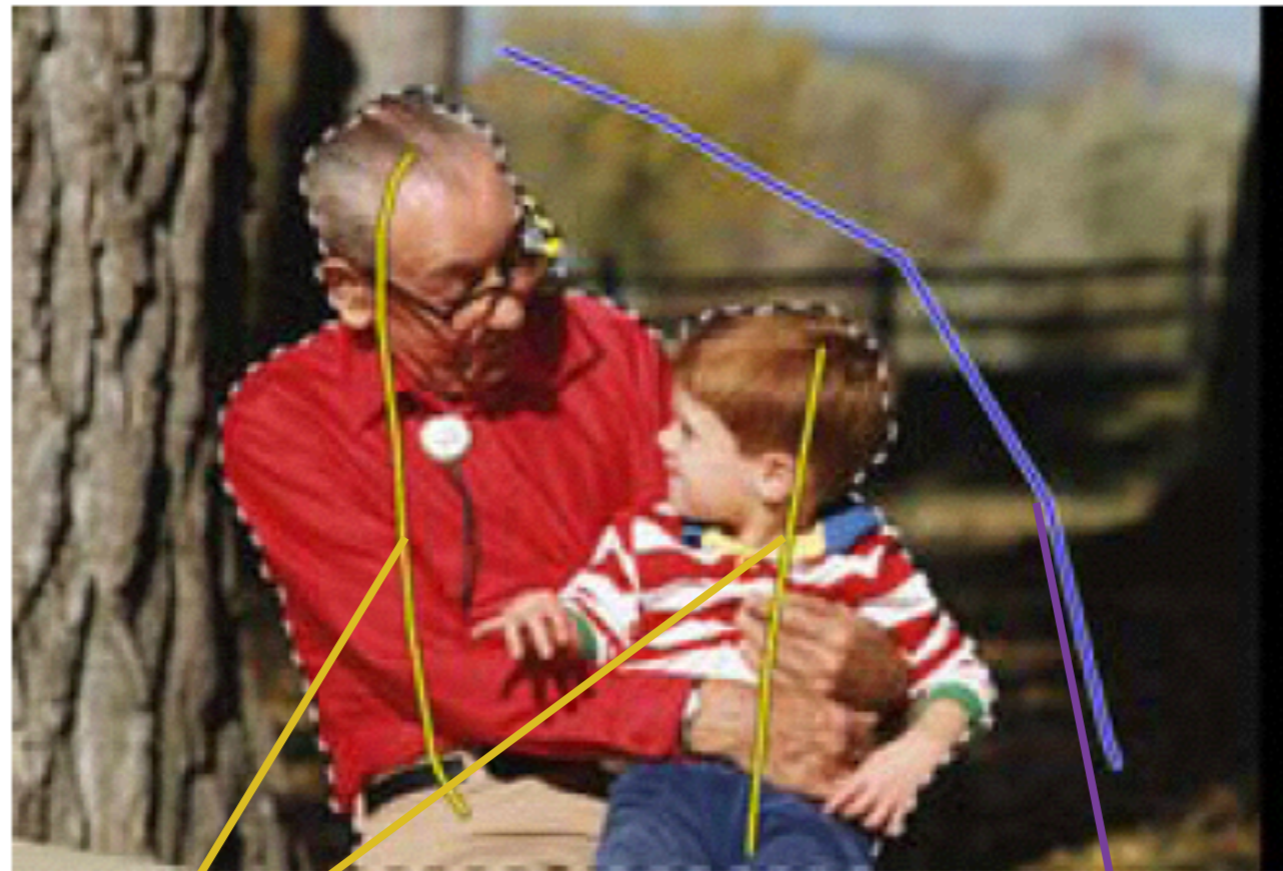


# Segmentation par coupure de graphe



# Modéliser les régions

- Modèle de couleur: basé sur la sélection de l'utilisateur

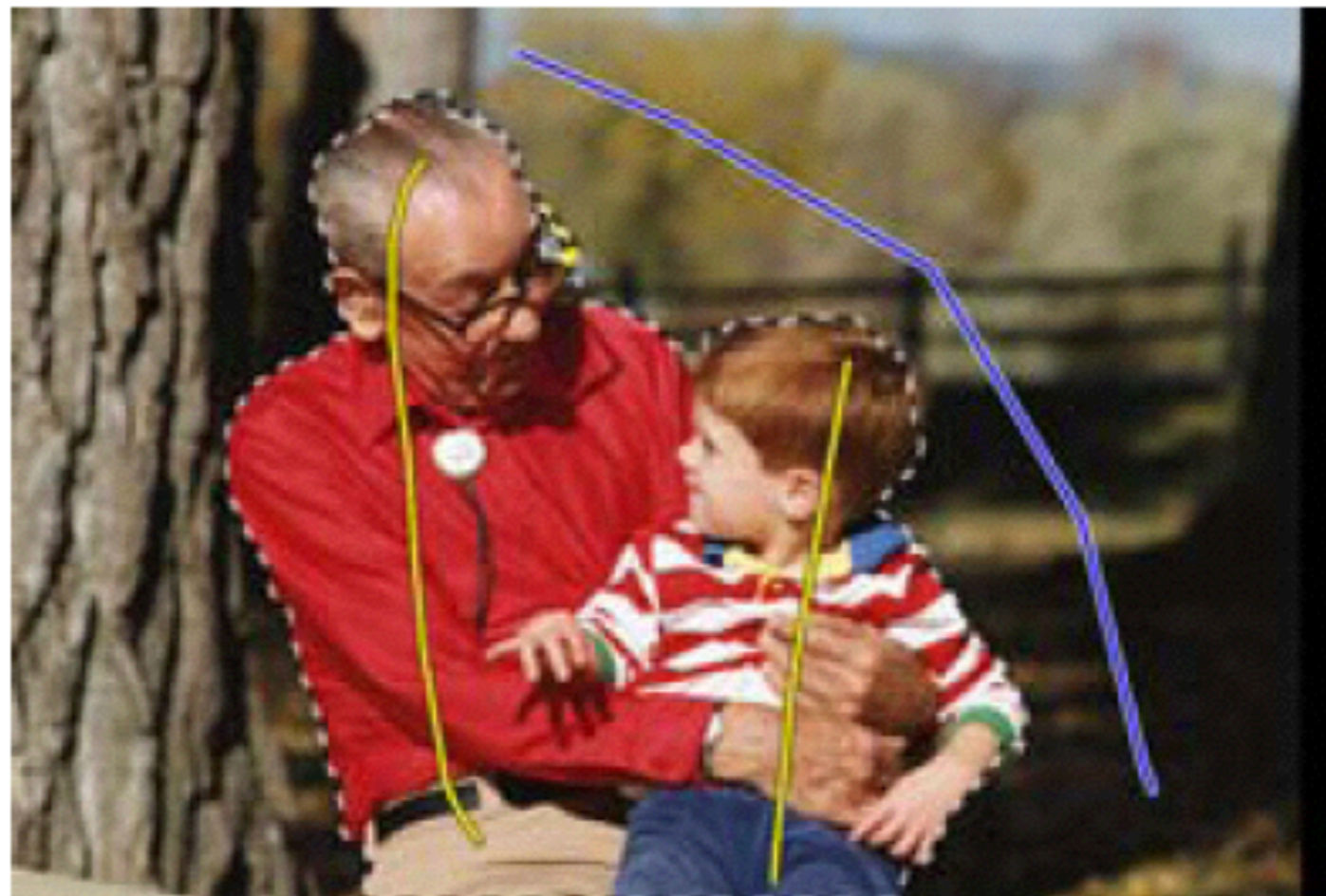


**Objet:** rouge, beige, blanc

**Arrière-plan:** noir, vert-jaune

# Modéliser les régions

- Probabilité qu'une couleur fasse partie
  - de l'objet  $P(c; \theta_{obj})$
  - de l'arrière-plan  $P(c; \theta_{ap})$
- Histogrammes, mixture de gaussiennes



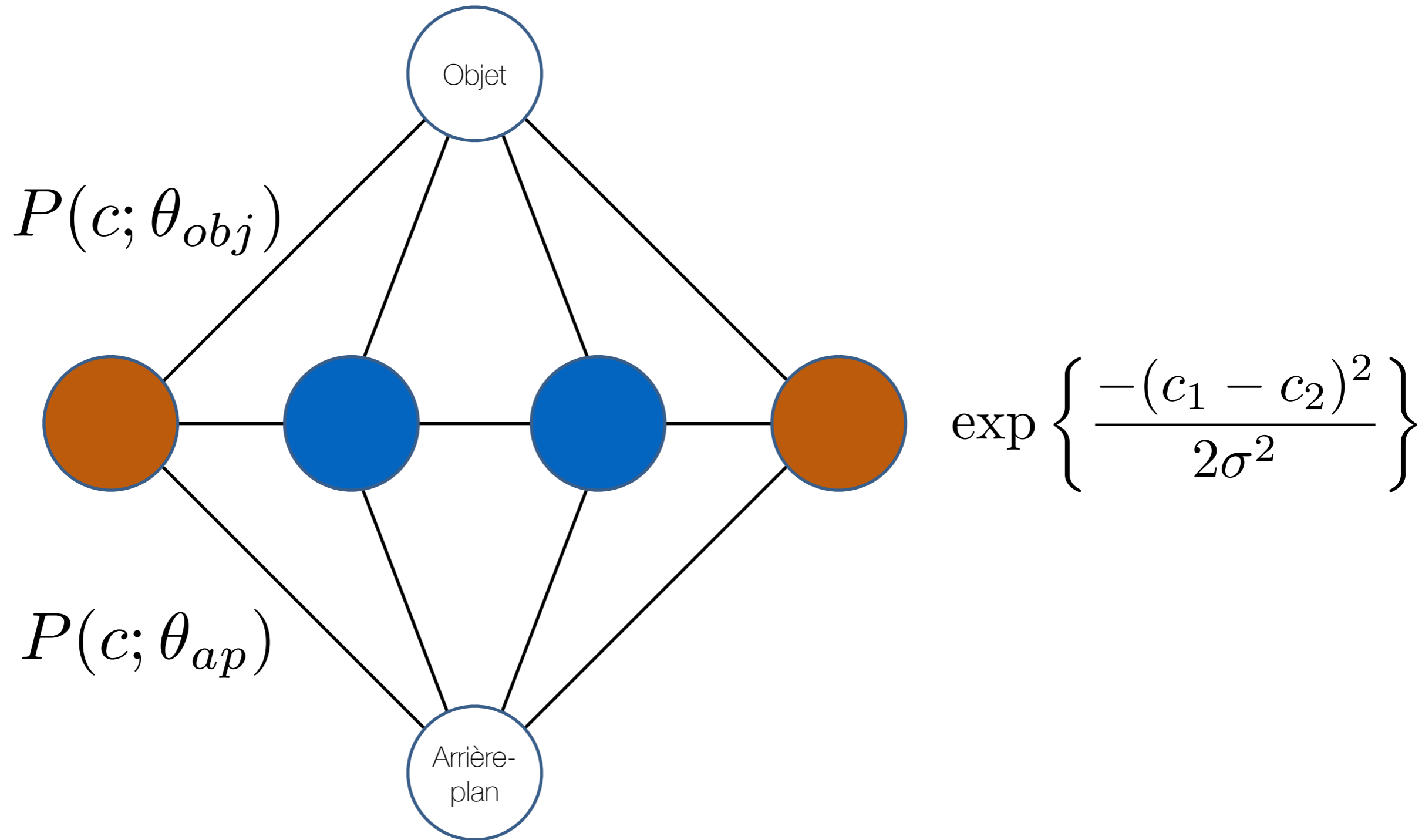
# Modéliser les frontières

- Similarité entre les couleurs de deux pixels adjacents

$$\exp \left\{ \frac{-(c_1 - c_2)^2}{2\sigma^2} \right\}$$



# Segmentation par graphe



# Segmentation par coupure de graphe

1. Définir graphe (4- ou 8-voisins)
2. Définir poids pour premier- et arrière-plan
  1. Histogramme de couleur (ou mixture de gaussiennes)

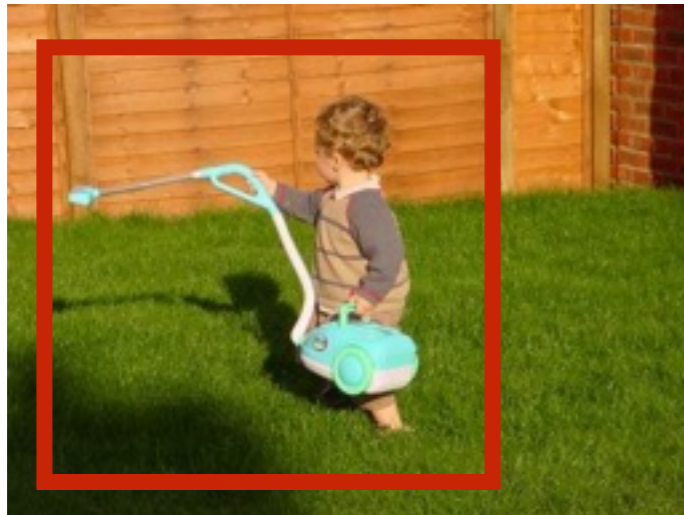
$$P(c; \theta_{obj}) \quad P(c; \theta_{ap})$$

3. Définir poids sur les arcs entre les pixels

$$\exp \left\{ \frac{-(c_1 - c_2)^2}{2\sigma^2} \right\}$$

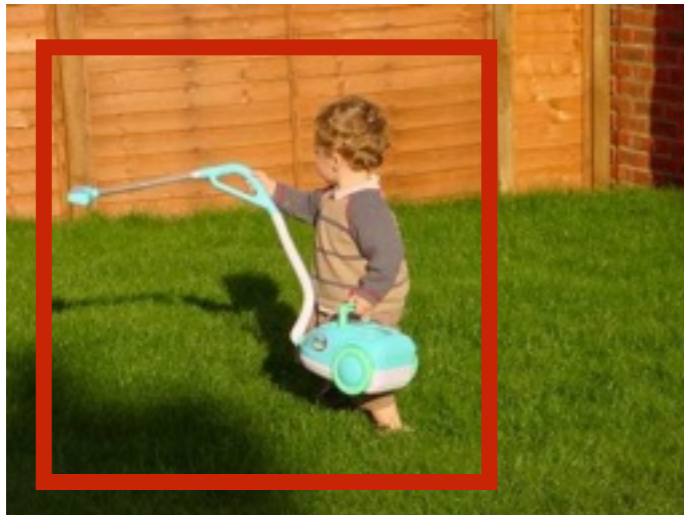
4. Appliquer algorithme de coupure minimale
5. Revenir à 2, mettre à jour les modèles de l'objet et de l'arrière-plan avec la nouvelle sélection

# Exemples faciles et difficiles



Pourquoi?

# Exemples faciles



# Exemples plus difficiles

Camouflage



Structure fine



Similarité avec  
arrière-plan



Démonstration (gimp)

# Limites des coupures de graphes

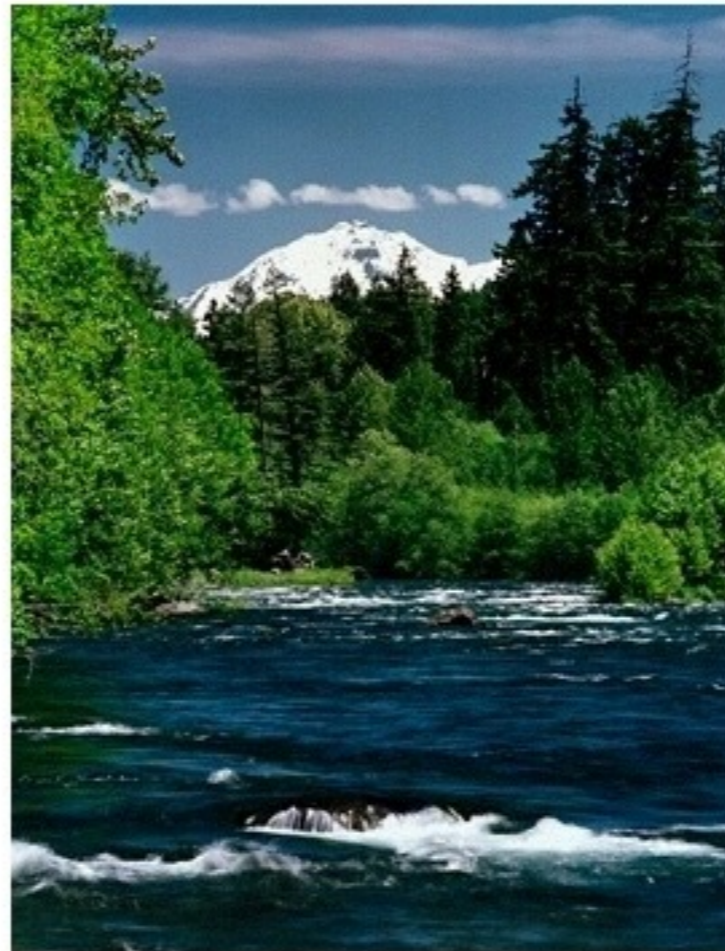
- Requiert des graphes associatifs
  - Noeuds connectés préfèrent avoir la même valeur
- Optimal seulement pour problèmes binaires (ex: objet vs arrière-plan)

# Autres applications: composition

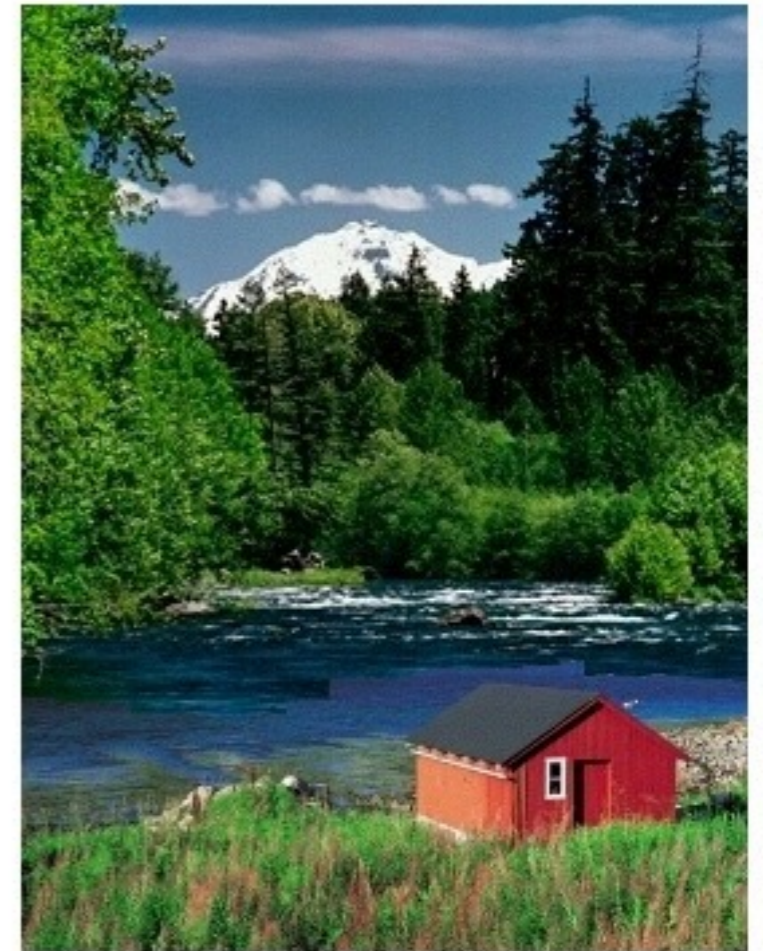
Graphcut Textures – Kwatra et al. SIGGRAPH 2003



+



=



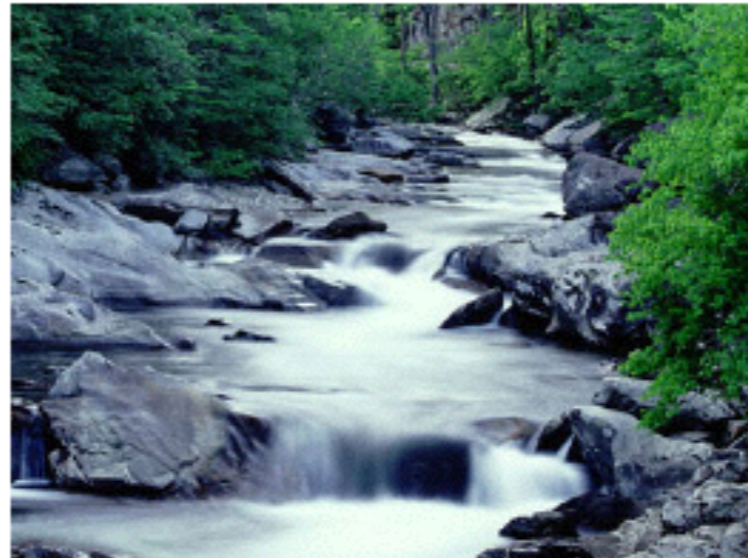


# Autre application: composition

Graphcut Textures – Kwatra et al. SIGGRAPH 2003



+



- Frontière idéale:
- Couleur similaire dans les deux images
- Gradient élevé dans les deux images



# Variante

“Soft Scissors” (Wang et al., SIGGRAPH'07)

<http://www.youtube.com/watch?v=M2Eh4yrlcg4>

# Résumé des idées principales

- Considérer l'image comme un graphe
  - Noeuds: pixels
  - Poids des arcs déterminé par les gradients entre les pixels
  - Parfois: rajouter des poids pour chaque pixel basé sur la ressemblance avec l'objet et l'arrière-plan
- Les bonnes frontières sont des chemins courts dans le graphe (Ciseaux intelligents, re-dimensionnement d'images)
- Les bonnes régions sont générées par une coupure avec coût minimum (coupure par graphe)