Traitement de l'image et du signal
Partie TI

Emanuel Aldea <emanuel.aldea@u-psud.fr>
http://hebergement.u-psud.fr/emi/453

Master Electronique, énergie électrique, automatique 1ère année

Estimation robuste

Cadre du problème :

- observations fournies par les images
  - points d'intérêt, contours, régions etc.
  - associations : appariements, champs de flot optique, etc.
- une partie importante des observations générée par un modèle mathématique défini par un ensemble $\boldsymbol{\theta}$ de paramètres

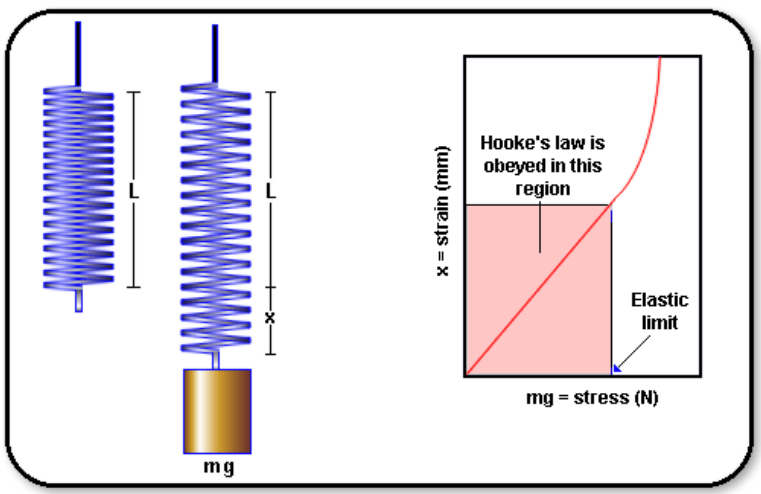
Objectif

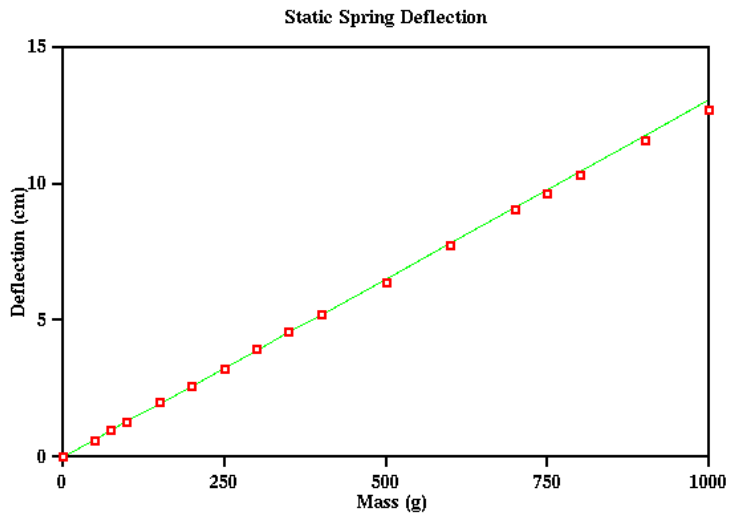
- déterminer les paramètres $\boldsymbol{\theta}$
  - en robotique : souvent une information de déplacement
  - suivi de cibles
  - état d'un système physique etc.
- nombre d'observations largement suffisant pour inférer $\boldsymbol{\theta}$ mais
- présence des valeurs aberrantes (outliers) qui ne respectent pas le modèle

Exemple jouet

La constante d'élasticité d'un ressort

- loi de Hooke : $F = kx$
- objectif : $\boldsymbol{\theta} = \{k\}$
  - on varie N fois la force appliquée, on mesure la déformation
  - N observations $\{(F_i, x_i)\}$
  - ensemble minimal de mesures pour déterminer $\boldsymbol{\theta}$: $K = 2$
  - en pratique on utilise les N observations pour une estimation par moindres carrés , à cause du bruit de mesure
- pas de outliers, tout s'explique par le modèle considéré





Exemple en vision

Estimation de l'ego-mouvement

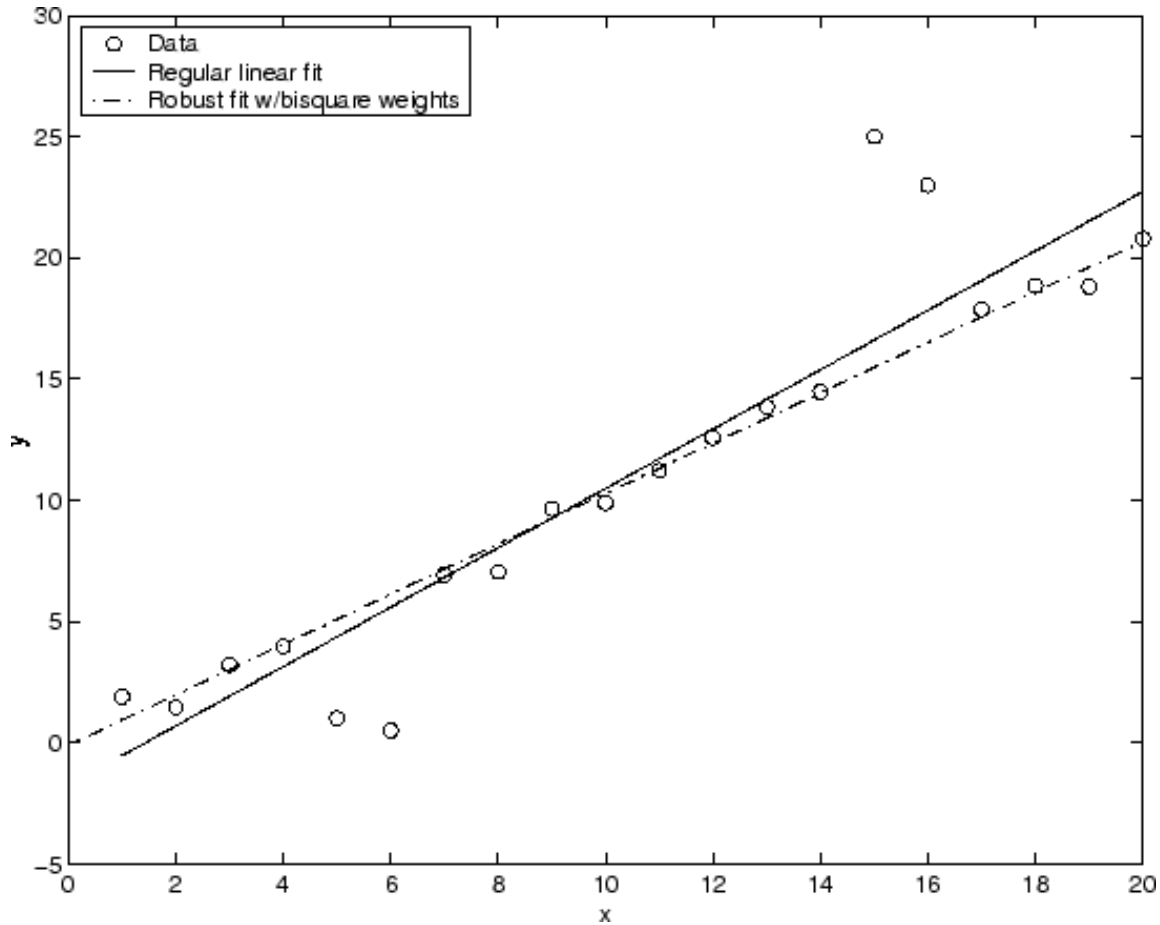
- N observations $\{x_i\}_{1 \leq i \leq N}$ (une obs. par pixel)
- ensemble minimal de taille K, $N \gg K$
- objectif : $\boldsymbol{\theta} = \{\boldsymbol{R}, \boldsymbol{t}\}$
- un algorithme f qui fournit $\boldsymbol{\theta} = f(x_1, \ldots, x_K)\}$
- problème : hypothèse de scène statique
- éléments dynamiques $\Rightarrow$ observations qui ne respectent pas le modèle $\boldsymbol{\theta}$

Objectif : déterminer $\boldsymbol{\theta}$ **et** les observations valides

Source du problème

Influence des outliers

- on ne peut pas ignorer les outliers et déterminer les paramétrés du modèle



- les méthodes de type moindres carrés sont très sensibles aux outliers à cause de la fonction d'erreur quadratique $\rho(r_i) = r_i^2$

Deux types d'approches

Analyse de l'ensemble des résidus

- Least Median of Squares (LMedS) ; on remplace la somme par la médiane des résidus :

$$\min_{\boldsymbol{\theta}} \operatorname{med} \rho(r_i)$$

- Least Trimmed Squares (LTS) ; tri des résidus et sélection des premiers $N/2 < M < N$

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{M} \rho(r_i)$$

- Recherche exhaustive nécessaire par K-tuples ; breakdown point $\sim 50\%$

Modification de ρ

Utilisation à la place de l'erreur quadratique d'une autre fonction symétrique, définie positive (voir Huber, Tukey etc.) Breakdown point inférieur à $1/K$

Dans tous les cas, on sépare les inliers et seulement après on peut appliquer une LMS classique.

RANSAC

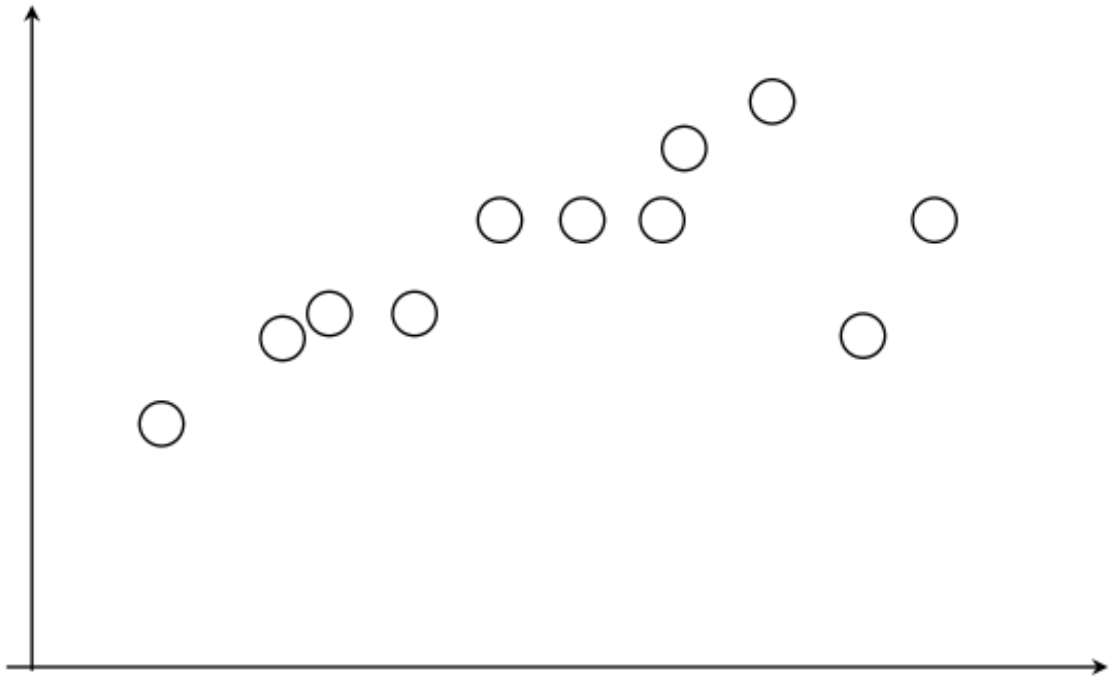
Random Sample Consensus

1. Pour T itérations / Tant qu'il reste du temps de calcul
   - sélection aléatoire de K observations
   - détermination exacte de $\boldsymbol{\theta}$
   - calcul cardinal du support pour $\boldsymbol{\theta}$: $\{x_i \quad t.q \quad \rho(x_i, \boldsymbol{\theta}) < \tau\}$
2. validation de $\hat{\boldsymbol{\theta}}$ ayant le plus grand support
3. calcul de $\tilde{\boldsymbol{\theta}}$ par moindres carrés sur le support de $\hat{\boldsymbol{\theta}}$

Paramètres

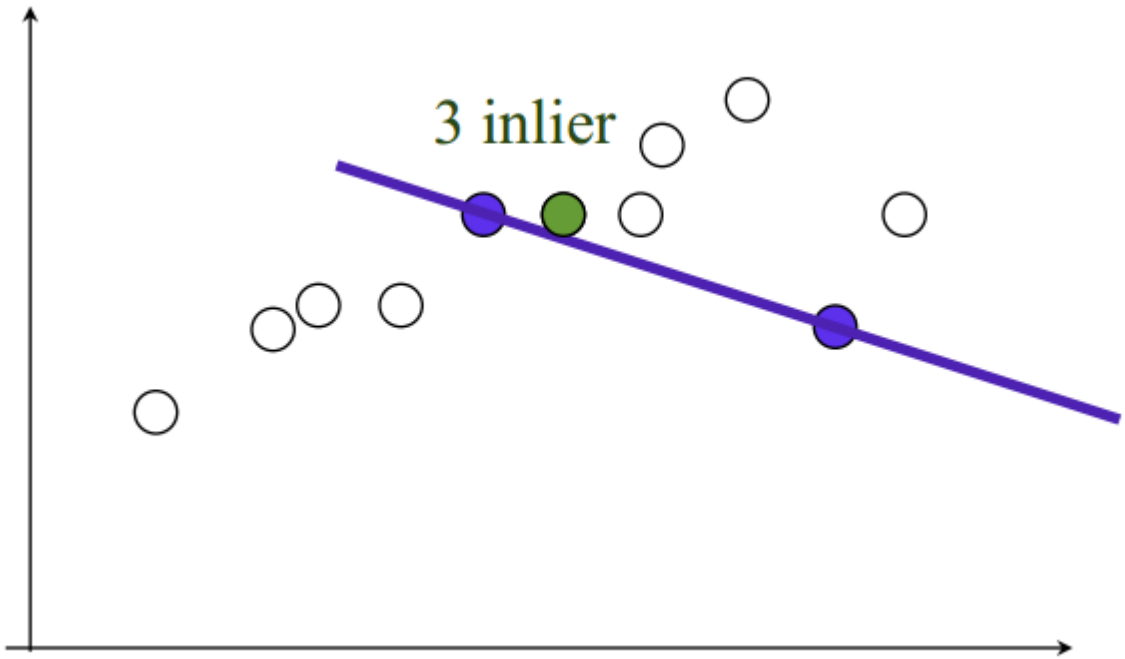
- τ pour l'inclusion dans le support
- le nombre de tirages P
- dépendent de l'application et de la proportion d'outliers

Exemple en 2D



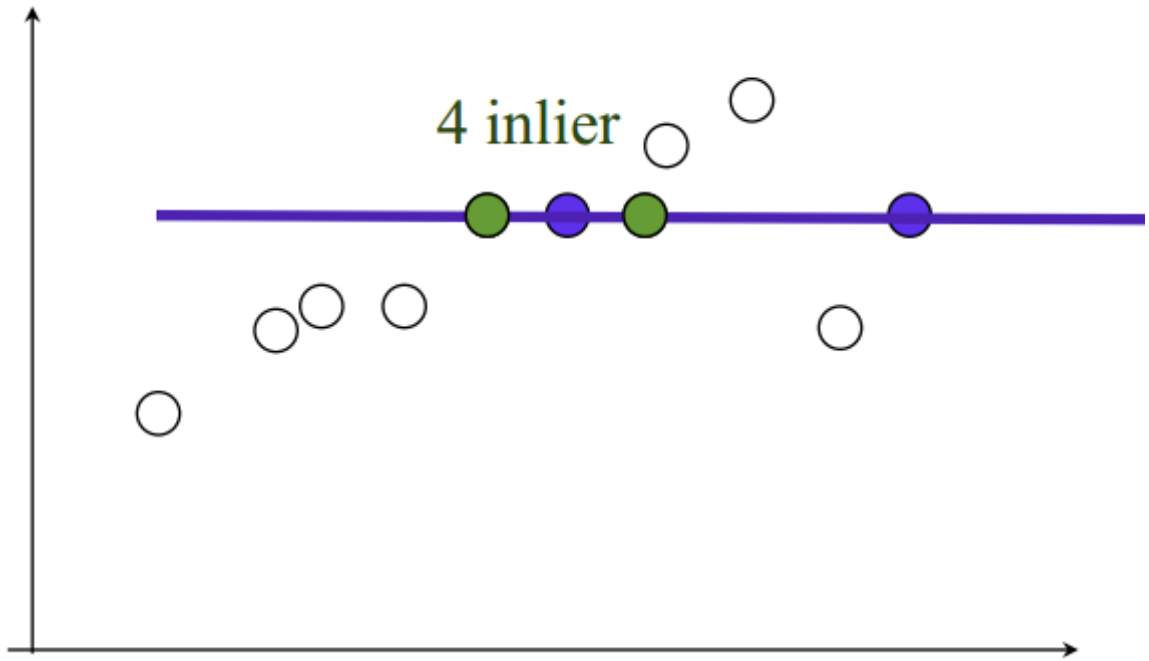
Ensemble initial

Exemple en 2D



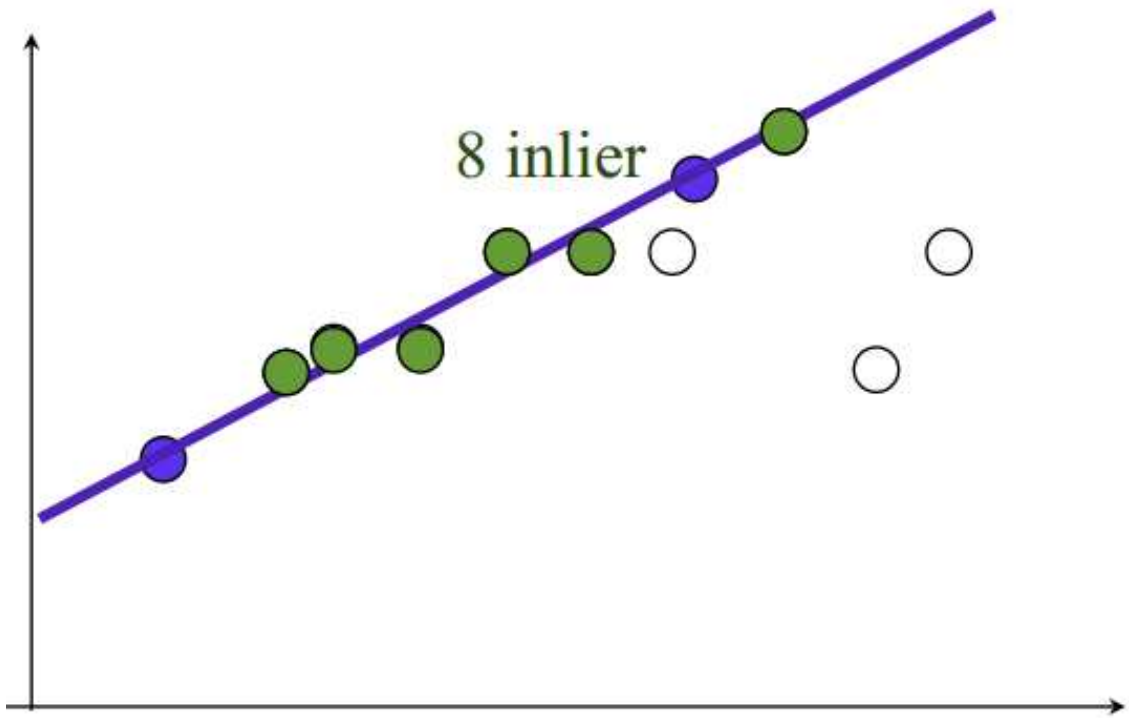
Fit line - 3 inliers

Exemple en 2D



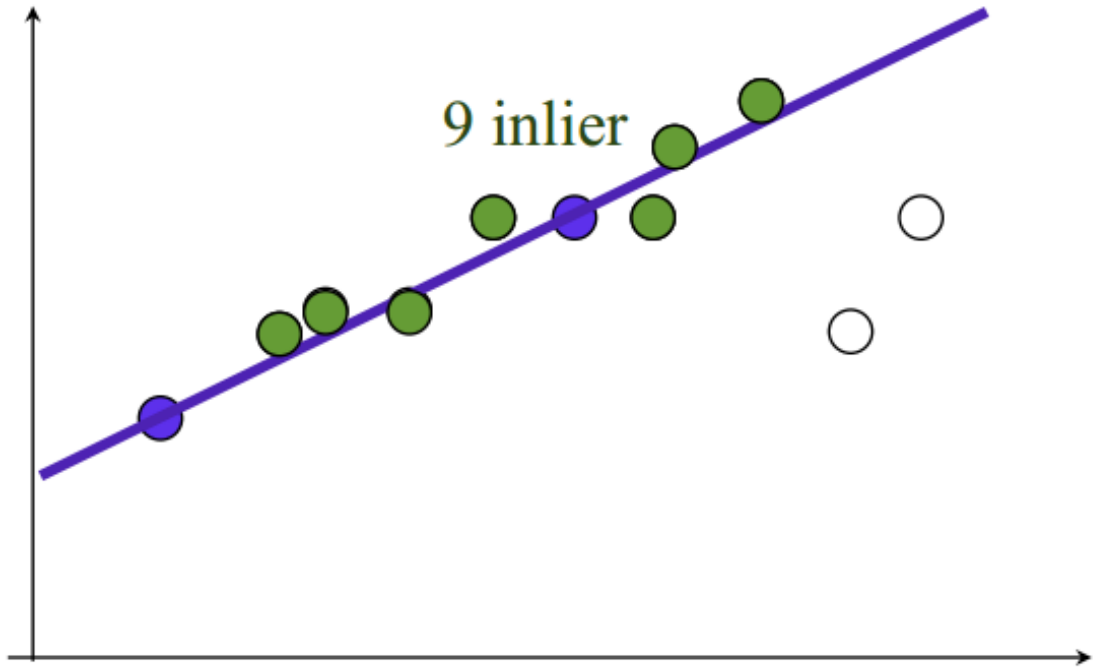
Fit line - 4 inliers

Exemple en 2D



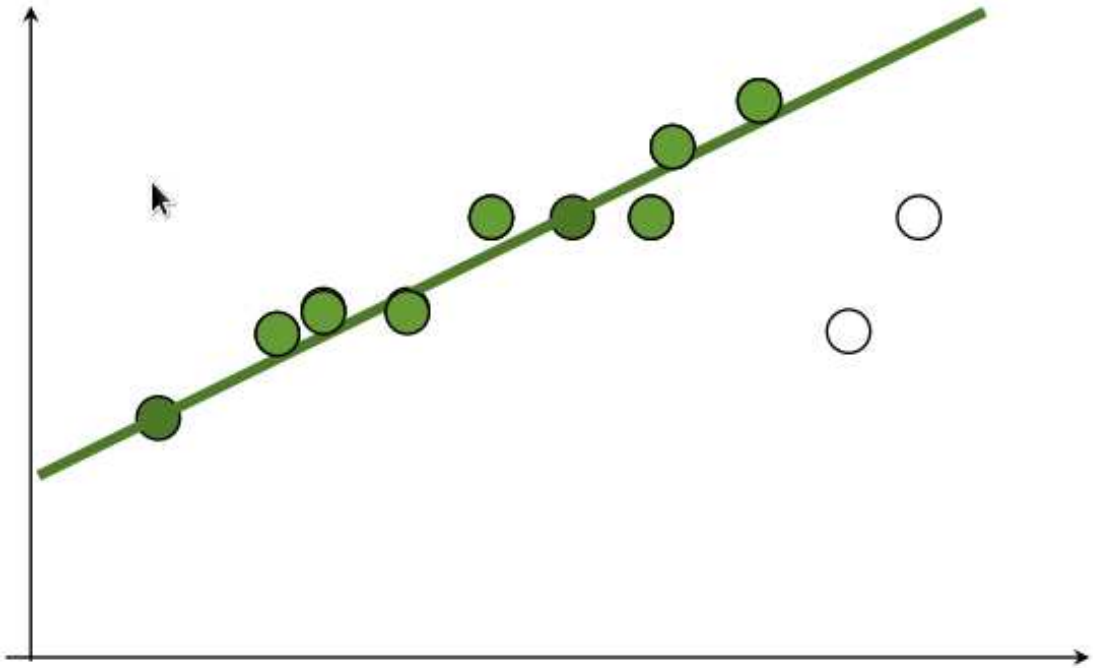
Fit line - 8 inliers

Exemple en 2D



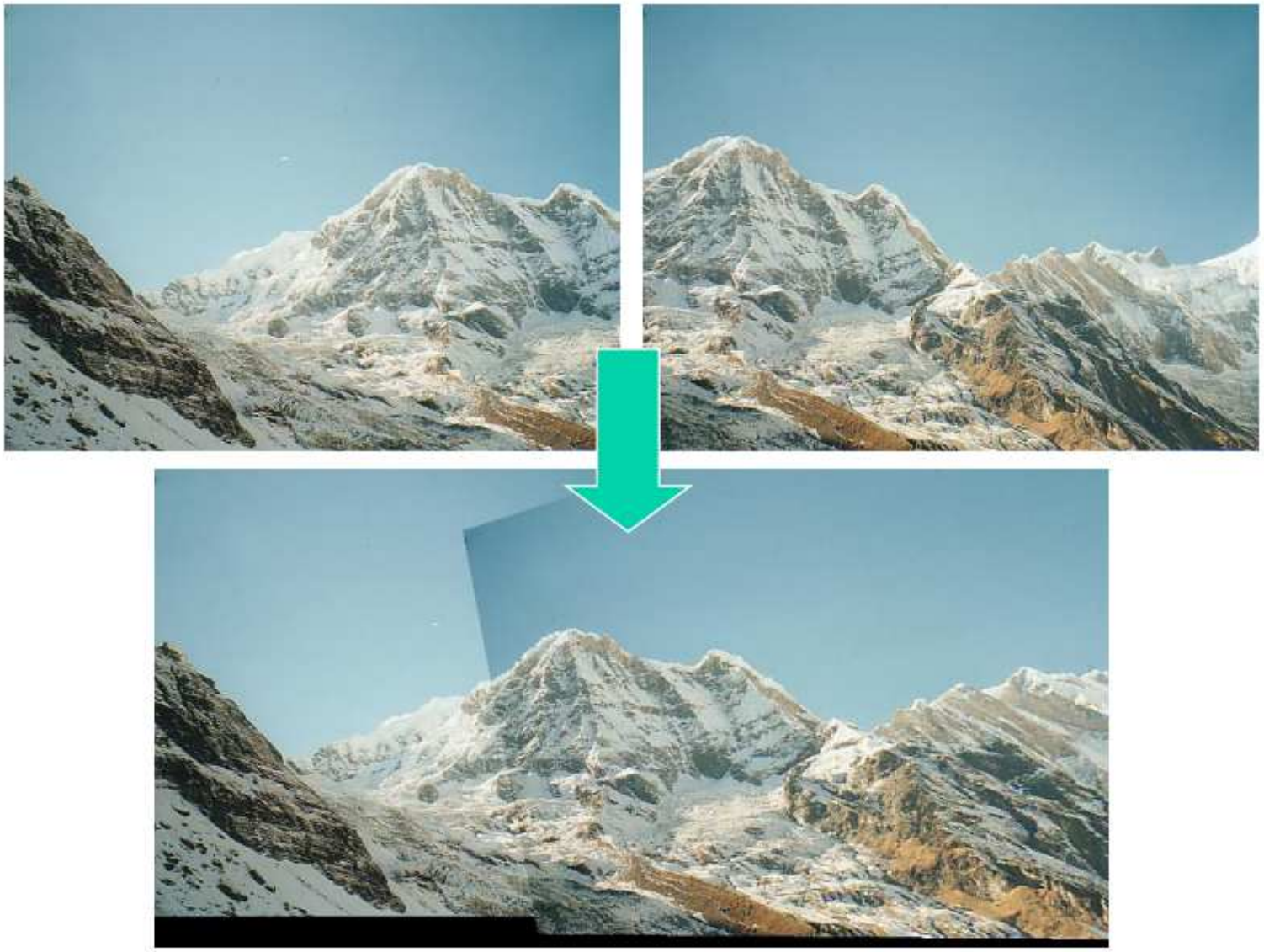
Fit line - 9 inliers

Exemple en 2D



Estimation finale par moindre carrés

Reconstruction panoramique



Objectif de l'opération

Reconstruction panoramique

Problème

- Détection de points d'intérêt et association
- Observation de type (x, y, x', y') : le coin (x, y) dans la première image est associé au coin (x', y') dans la deuxième image
- si rotation pure de la camera entre les deux images $\tilde{\mathbf{x}}' = \mathbf{H}\tilde{\mathbf{x}}$ ou :

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- en développant

$$\begin{cases} x' &= \frac{h_{00}x+h_{01}y+h_{02}}{h_{20}x+h_{21}y+h_{22}} \\ y' &= \frac{h_{10}x+h_{11}y+h_{12}}{h_{20}x+h_{21}y+h_{22}} \end{cases}$$

Reconstruction panoramique

Problème

- les inconnues étant les différents h_{ij}

$$\begin{cases} x'(h_{20}x + h_{21}y + h_{22}) &= h_{00}x + h_{01}y + h_{02} \\ y'(h_{20}x + h_{21}y + h_{22}) &= h_{10}x + h_{11}y + h_{12} \end{cases}$$

$$\begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -x'x & -x'y & -x' \\ 0 & 0 & 0 & x & y & 1 & -y'x & -y'y & -y' \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Reconstruction panoramique

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1x_1 & -y'_1y_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -x'_nx_n & -x'_ny_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -y'_nx_n & -y'_ny_n \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \end{bmatrix} = \begin{bmatrix} x'_1 \\ y'_1 \\ \vdots \\ x'_n \\ y'_n \end{bmatrix}$$

H est déterminée modulo un facteur multiplicatif, donc on peut fixer h_{22} à 1. On retient donc que pour le problème d'estimation de l'homographie $K = 4$. Il faut donc résoudre $\mathbf{Ah} = \mathbf{b}$.

Si $n > 4$ le système est surdéterminé. En général pour trouver la solution par moindres carrés de $\mathbf{Ah} = \mathbf{b}$ il faut :

1. calculer la décomposition en valeurs singulières (la SVD) de $\mathbf{A}$: $\mathbf{A} = \mathbf{UDV}^T$
2. calculer $\mathbf{b}' = \mathbf{U}^T\mathbf{b}$
3. trouver $\mathbf{y}$ defini par $y_i = b'_i/d_i$
4. la solution est $\mathbf{h} = \mathbf{Vy}$

Définition de la segmentation

- Subdiviser l'image en structures :
  - des objets,
  - des régions
  - $\cup_{i=1}^{N} R_i = X$
  - $R_i \cap R_j = \emptyset$
  - $\forall i, R_i$ est connexe
- Objets : une forme particulière, par exemple :
  - des points,
  - des segments de droites,
  - des contours,
  - des formes plus complexes

Définition de la segmentation

- Régions : pas d'hypothèse particulière sur la forme mais des propriétés sur la distribution des niveaux de gris supposés vérifiées en tout point de la région, par exemple :
  - homogénéité en niveau de gris,
  - texture particulière.
- des régions qui vérifient à la fois des critères de forme et de texture.
- le Graal du traitement de l'image : un sujet difficile, une méthode (imparfaite) traite un cas particulier.

Qu'est-ce que c'est une bonne segmentation ?

Qu'est-ce que c'est une bonne segmentation ?

a : 4 régions

b : 16 régions

c : 20 régions

d : 41 régions

e : 79 régions

Segmentation par découpage

- Construire une représentation en *Quadtree* de l'image.

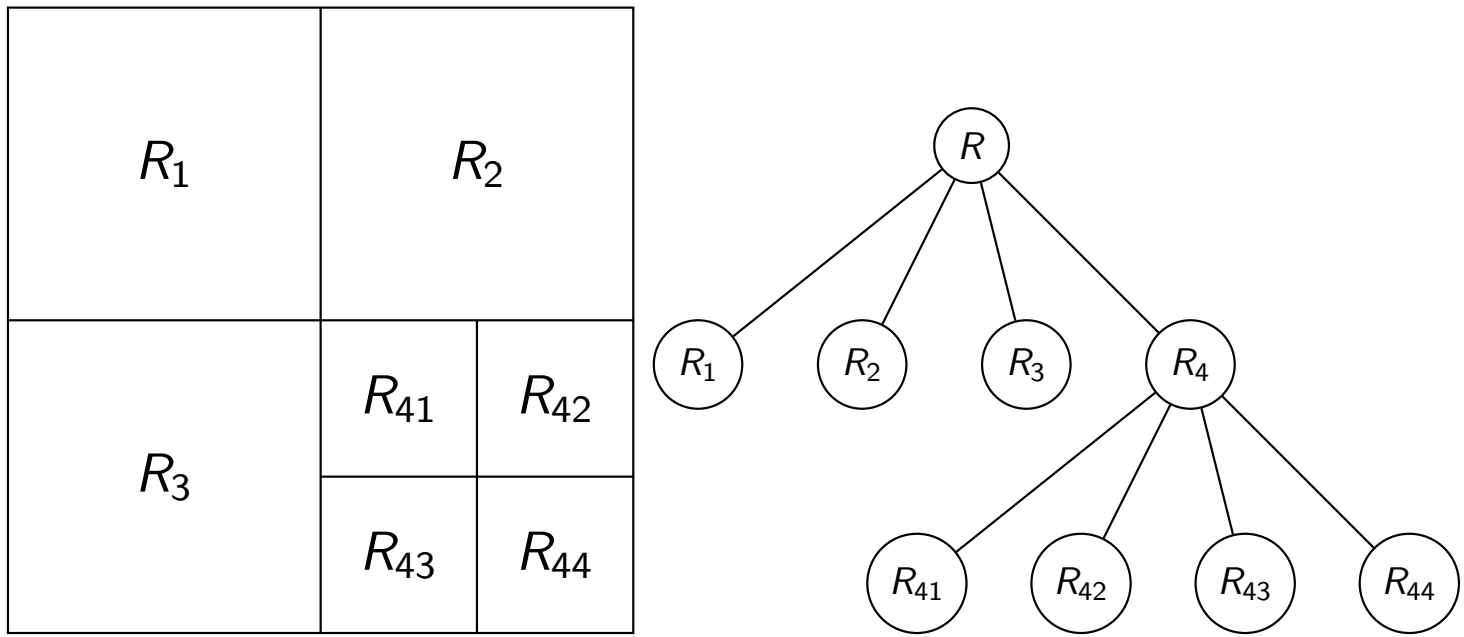


FIGURE – Décomposition en *Quadtree*

- Dans chaque région, un critère d'homogénéité de la répartition des niveaux de gris (ou des couleurs) est respectée.

Segmentation par découpage

- Pour décider du découpage d'une région R en 4 sous-régions, on se donne un critère (à préciser ultérieurement), appelé aussi prédicat, qui décide si une région est homogène ou non.
- Soit un algorithme

```
algo pred(R:region): booleen;
```

qui retourne `VRAI` lorsque le critère d'homogénéité est vérifié pour la région R, et `FAUX` sinon.

- On cherche à découper l'image en régions R_i telles qu'à la fin de l'algorithme on ait :
  1. $I = \bigcup_i R_i$ et $R_i \bigcap R_j = \emptyset$
  2. `Pred(`R_i`) = VRAI`

Segmentation par découpage

- Un tel algorithme est facilement implanté récursivement :

```
algo split_rec( I:image, R:region, qt: arbre 4-aire)
   si pred(R) = FAUX et taille(R) > rmin alors
      qt.no := creer_noeud();
      qt.ne := creer_noeud();
      qt.so := creer_noeud();
      qt.se := creer_noeud();
      split_rec(I,R.nord_ouest,qt.no);
      split_rec(I,R.nord_est,qt.ne);
      split_rec(I,R.sud_ouest,qt.so);
      split_rec(I,R.sud_est,qt.se);
   fin si
fin algo
```

Segmentation par découpage

```
algo split( I:image): arbre 4-aire
    racine := creer_noeud();
    split_rec( I, dims(I), racine);
    split := racine;
fin algo
```

- En pratique, chaque noeud qt de l'arbre doit contenir les informations suivantes :
  - coordonnées de la région (un rectangle),
  - statistiques diverses sur la région image, utile pour calculer le critère de découpage.

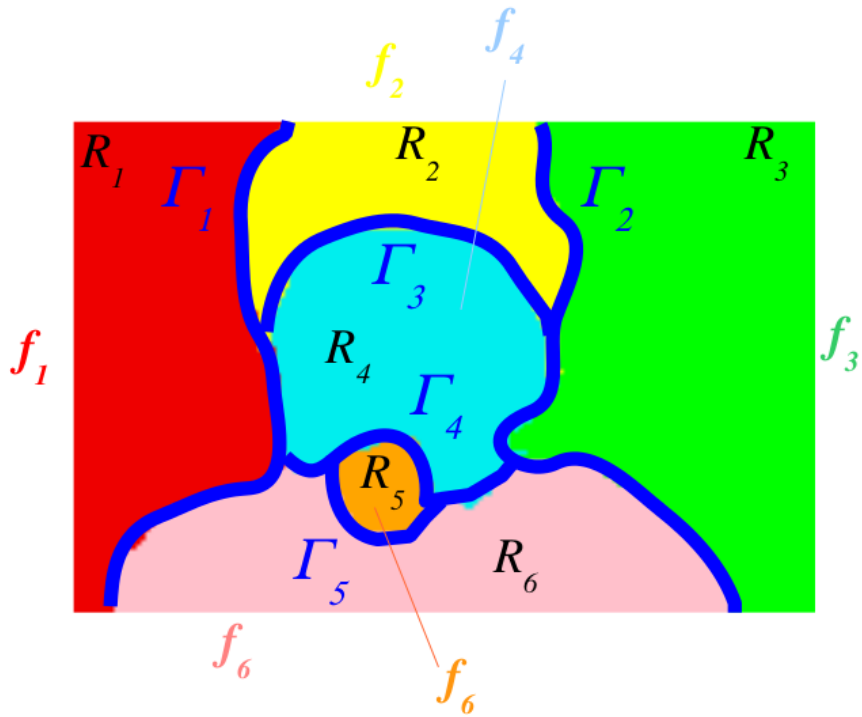
Segmentation par découpage

- Dépendra des propriétés que l'on souhaite voir vérifiées par chaque région après segmentation.
- Homogénéité des niveaux de gris :
  - après segmentation chaque région est homogène dans sa distribution des valeurs de niveaux de gris : la variance de la région est plus petite qu'un seuil à fixer.
  - Remarque : une région réduite à un pixel est de variance nulle.
- Homogénéité en texture/couleurs : non abordée dans ce cours d'initiation.
- Taux de contours réduit : appliquer un détecteur de contours dans la région R et découper si :

$$\tau = \frac{|\text{points de contours}|}{|R| - |\text{points de contours}|} > \text{seuil}$$

- Critère mixte (homogénéité, taux de contours).

Segmentation par optimisation



Determination d'une fonction f bidimensionnelle, constante par morceaux, proche de l'image I, avec une partition simple dpdv. géométrique. Fonctionnelle de coût :

$$K = \sum_i \iint_{R_i} (I(x,y) - f_i)^2 dxdy + \mu \sum_j \int_{\Gamma_j} dl$$

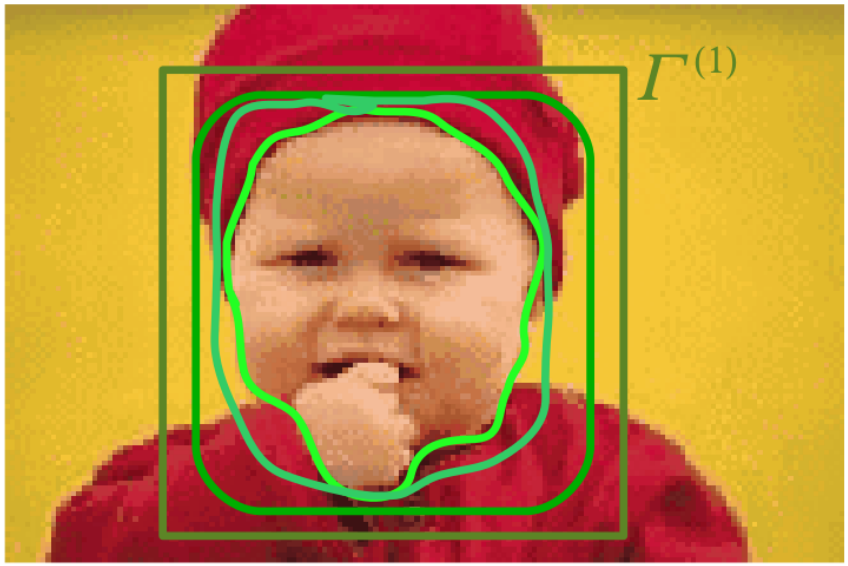
Segmentation par optimisation

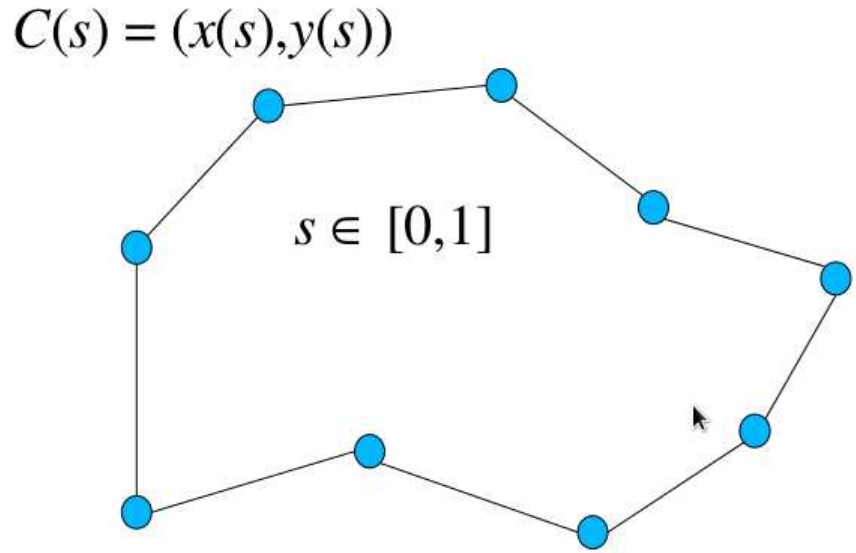
$$K = \sum_i \iint_{R_i} (I(x,y) - f_i)^2 dxdy + \mu \sum_j \int_{\Gamma_j} dl$$

Pas de solution directe au problème de minimisation. Il y a deux techniques pour approcher la solution

- méthodes variationnelles sur des courbes fermées
- méthodes markoviennes par itération à partir d'une segmentation initiale

Active contours

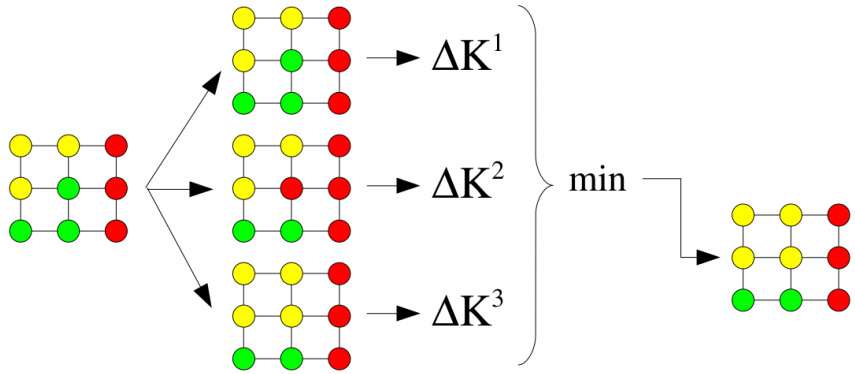




Fonctionnelle de coût $E = E_i + E_e$ où :

- $E_i = \int_0^1 \alpha(C'(s))^2 ds$ pénalise la longueur du snake
- $E_e = \int_0^1 -\nabla I(x(s), y(s)) ds$ favorise l'alignement sur les forts gradients

Techniques markoviennes



Fonctionnelle de coût

$$K = \sum_i \iint_{R_i} (I(x,y) - f_i)^2 dxdy + \mu \sum_j \int_{\Gamma_j} dl$$

- partir d'une segmentation préalable (sur-segmentation)
- ajuster localement les labels de chaque pixel de façon à diminuer la fonction de coût