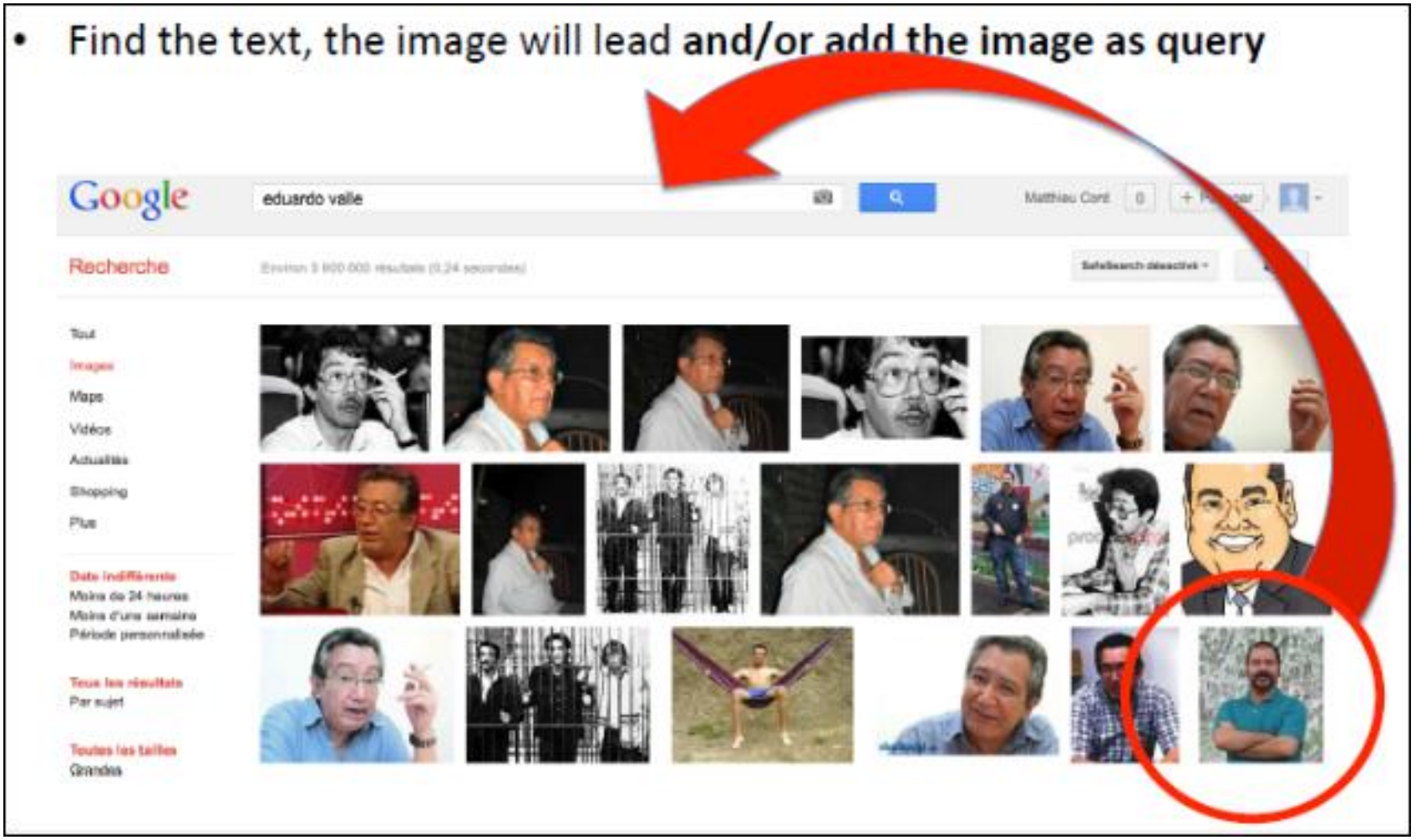
Indexation et extracteurs de points d’interet

Chapitre 5

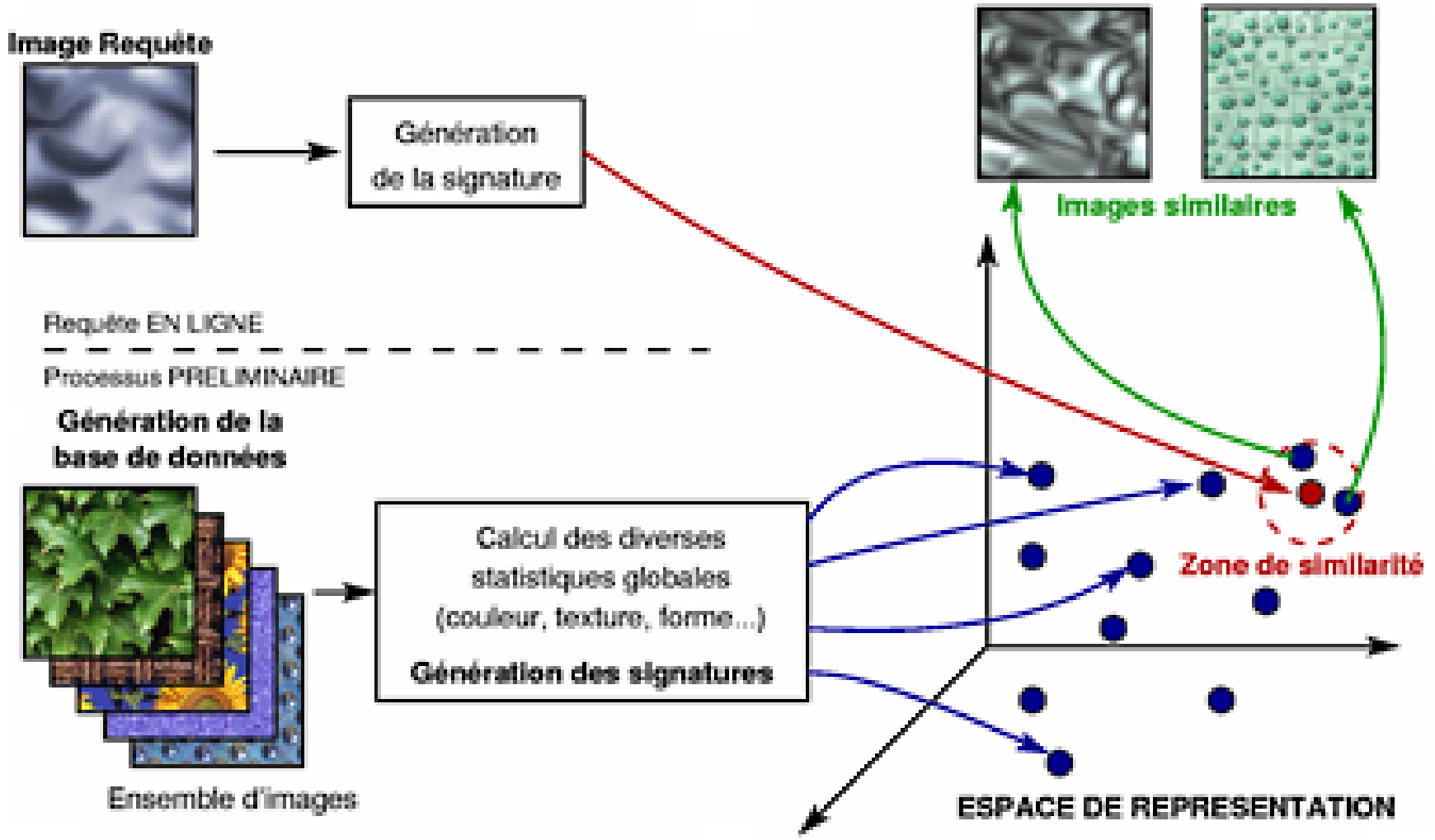
Big data : Rechercher des images…

- Rechercher visuelle plutôt que textuelle



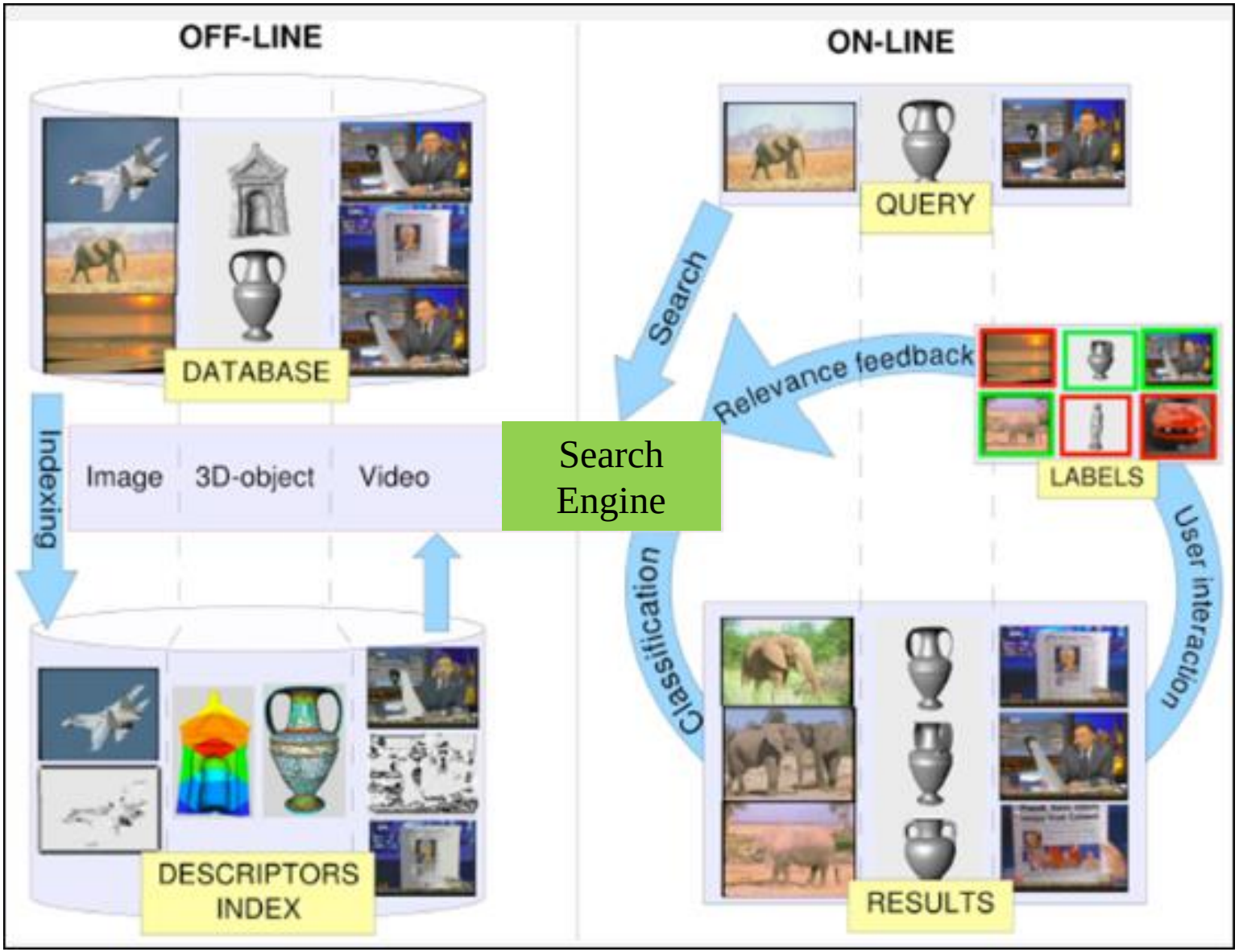
Décrire et comparer pour indexer…

- Comment caractériser une image ou région d'image ?



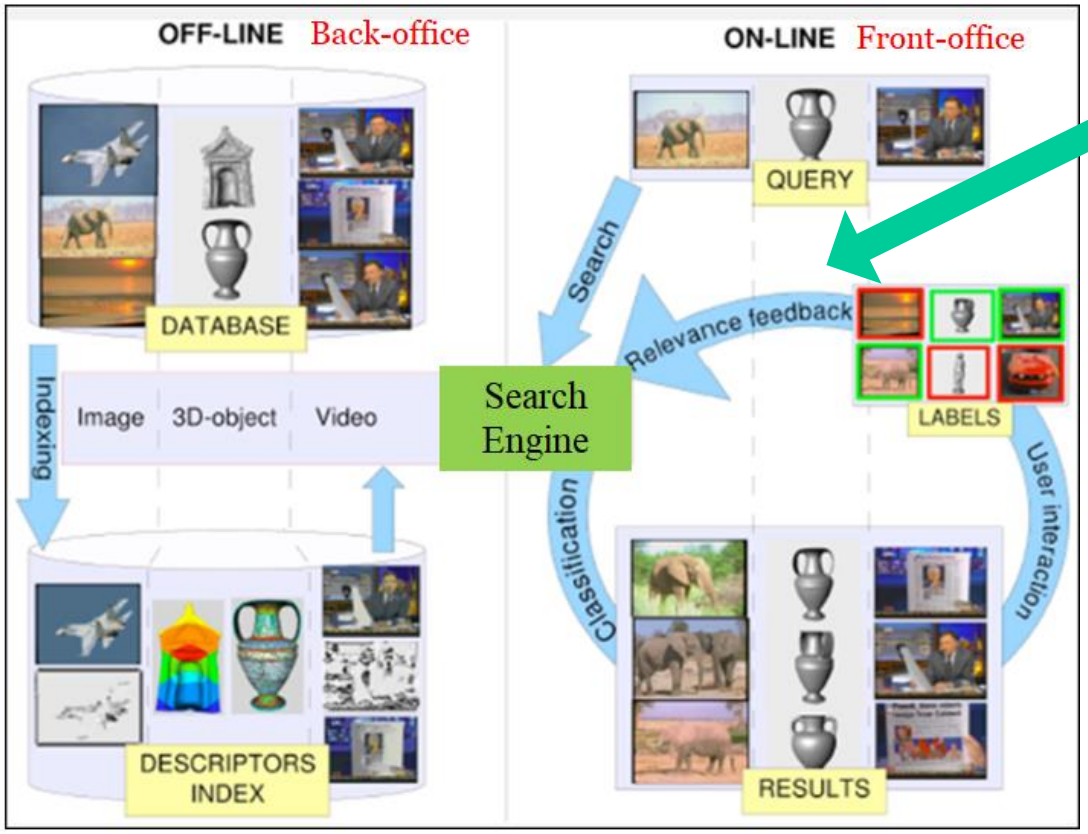
Décrire et comparer pour indexer…

Architecture ➔



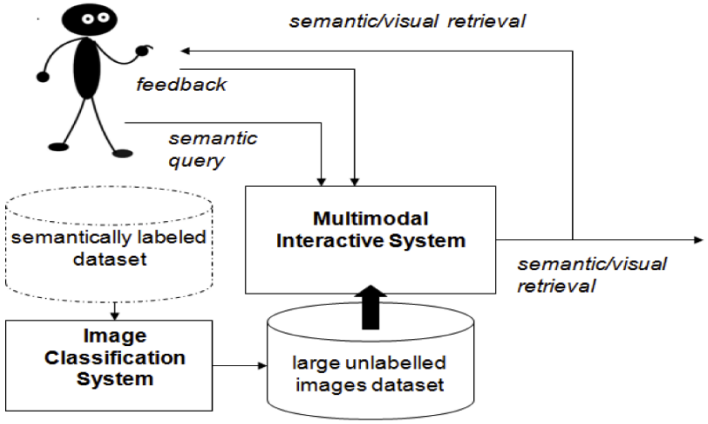
Décrire et comparer pour retrouver…

Méthode de “Relevance feedback”



- Impliquer l’utilisateur pour affiner une recherché d’images
- Amélioration de la méthode de Rocchio

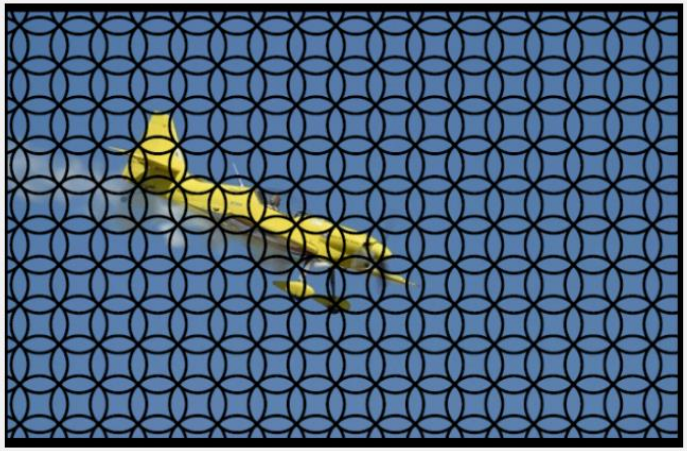
$$\overrightarrow{Q_m} = (a \cdot \overrightarrow{Q_o}) + \left(b \cdot \frac{1}{|D_r|} \cdot \sum_{\overrightarrow{D_j} \in D_r} \overrightarrow{D_j}\right) - \left(c \cdot \frac{1}{|D_{nr}|} \cdot \sum_{\overrightarrow{D_k} \in D_{nr}} \overrightarrow{D_k}\right)$$



Décrire et comparer les images

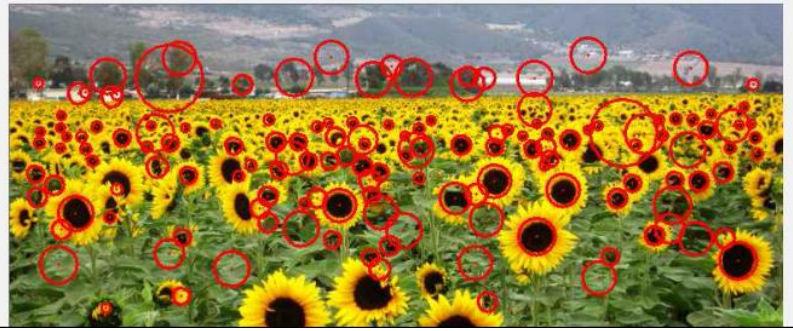
Primitives

- Méthodes denses, non basées sur le contenu visuel :
  - Pixels, grille dense d'échantillonnage



Primitives

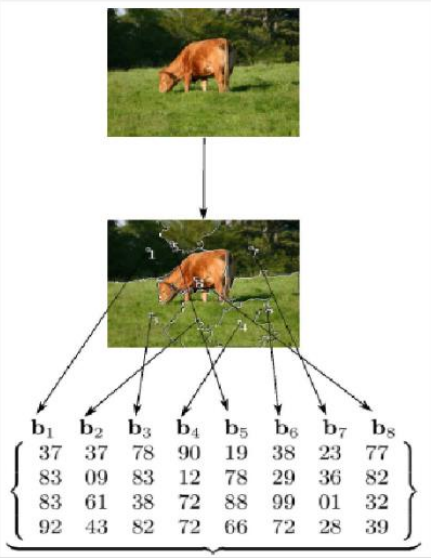
- Méthodes éparses (sparse), basées sur le contenu visuel :
  - Points d'intérêt (image ou vidéo), segmentation en régions



Signature = descripteurs

- Couleur : histogrammes RGB, HSV, *etc*
- Forme : Shape context, descripteurs de Fourier, *etc*
- Texture : Filtres de gabor *etc*
- Gradients : SIFT, SURF, HoG, *etc*
- Mouvement : HoF, STIP
- etc

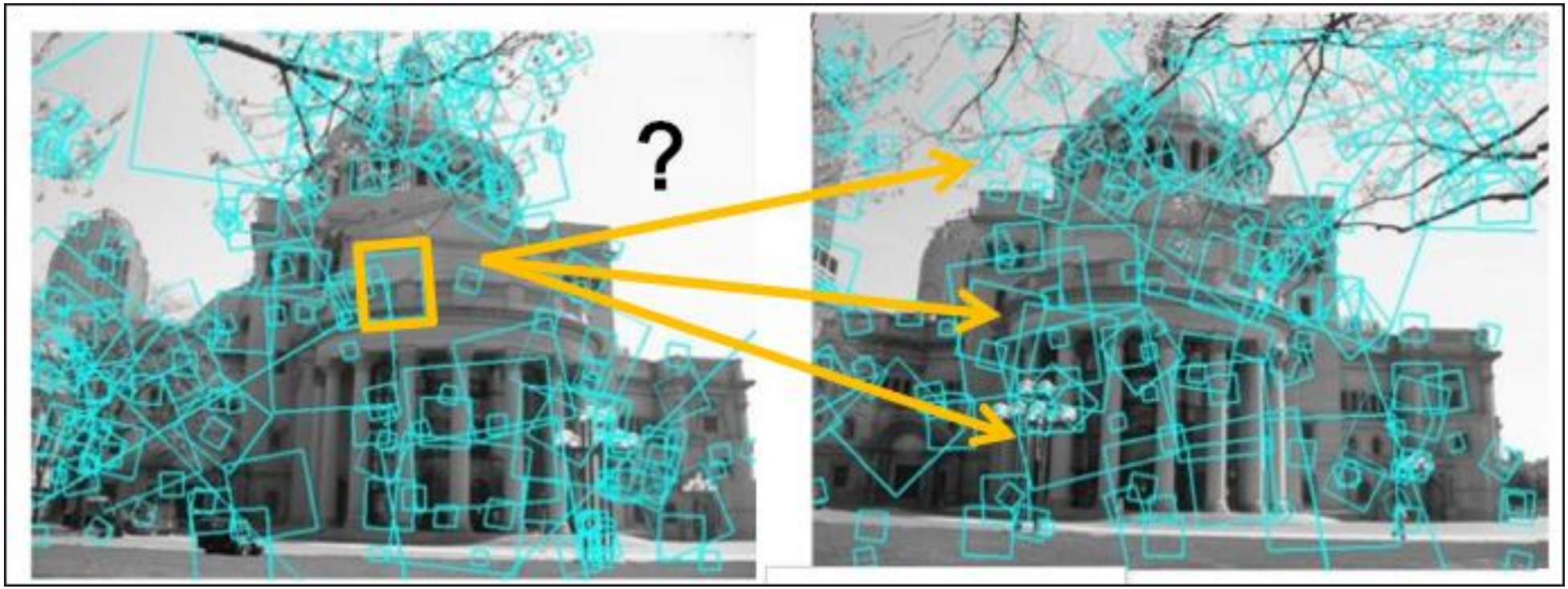
Attributs



- Couleur
- Forme
- Texture
- Gradients
- Mouvement
- etc

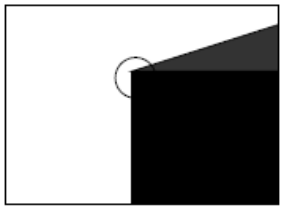
Décrire et comparer les images

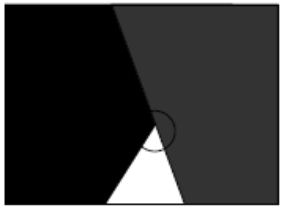
- Il devra être possible de faire des mises en correspondance locale



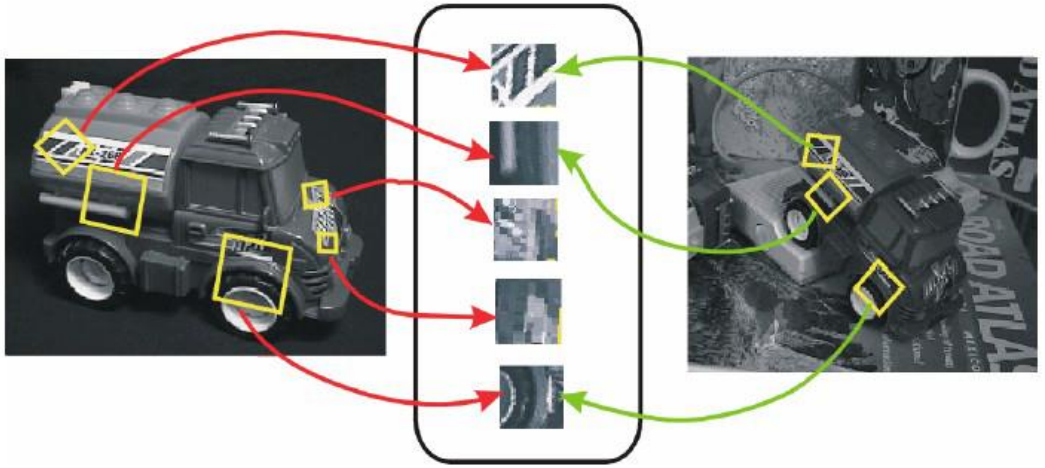
Qu’est ce qu’un point d’intéret ?

- Dépasser la notion de contours
  - Éviter la segmentation souvent source d’erreurs → Trouver des **Points Saillants**
  - 1 point contour : discontinuité de la fonction d’intensité dans une direction
  - I point d’intérêt : dans au moins deux directions ➔ Coins (corner)





- Propriétés recherchées
  - Localité
  - Robustesse
  - Discriminance
  - Quantité : assez mais pas trop
  - Efficacité : temps, précision
  - Invariance : rotation, échelle, photométrie

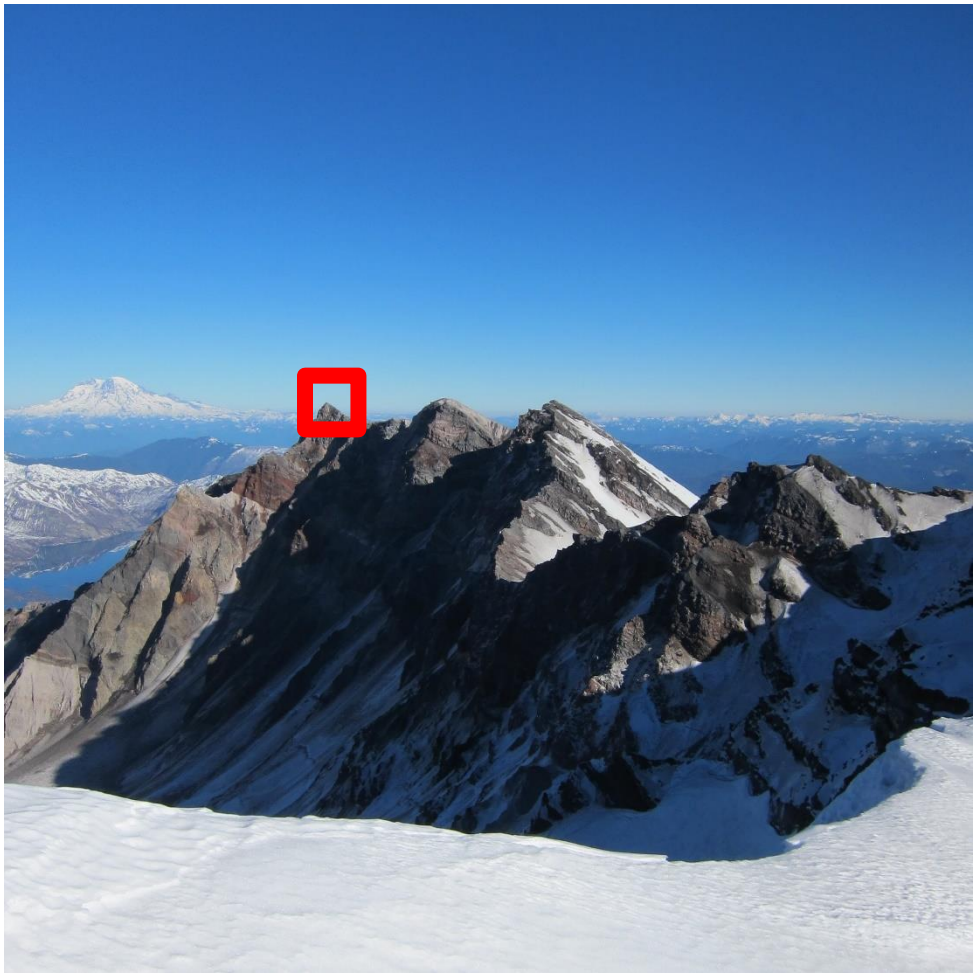


Trouver les bons points d’interet ?

- Exemple de la creation de pamorama
- Mise en correspondance de points d’interet
- Correspondance unique recherchée !

Trouver les bons points d'interet ?

- Zone unie (ciel) → Pas bon
  - Pas de variation
  - Correspondance multiple
- Contours → ok
  - Variation dans une direction
  - Correspodance le long du contour
- Coins → Good!
  - 1 seule correspodance



Trouver les bons points d'interet ?

- Calcul d'une "Energie" en chaque point:
  - Energie = Calcul de différences / distances au carré
  - Comparaison de l'image (plutôt d'une sous partie de l'image) avec elle même après un léger décalage

→ Auto-correlation

- Avec ou sans pondération

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

Voisinage considéré

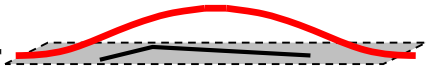
Intensité voisine

Intensité

voisinage $w(x,y)$ =

1 in window, 0 outside

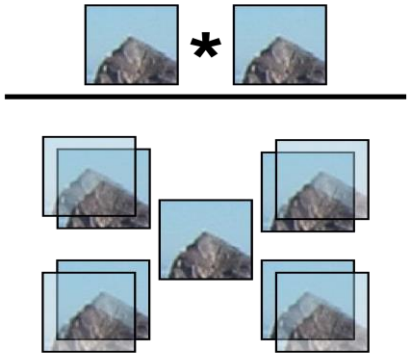
or

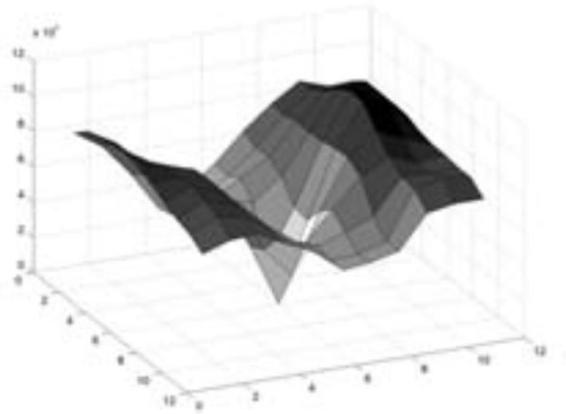


Gaussian

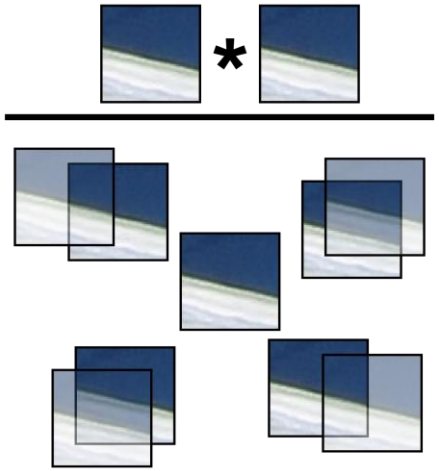
Energie ➔ valeurs de Self-difference

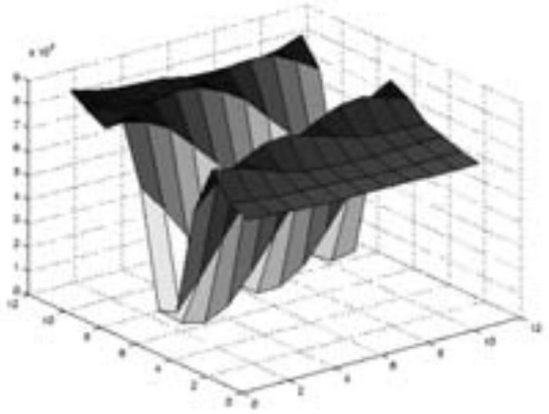
Corner: forte partout sauf sur le point



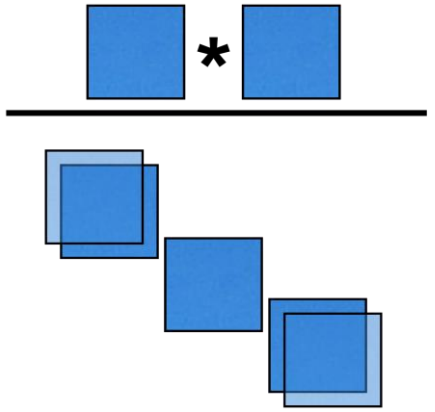


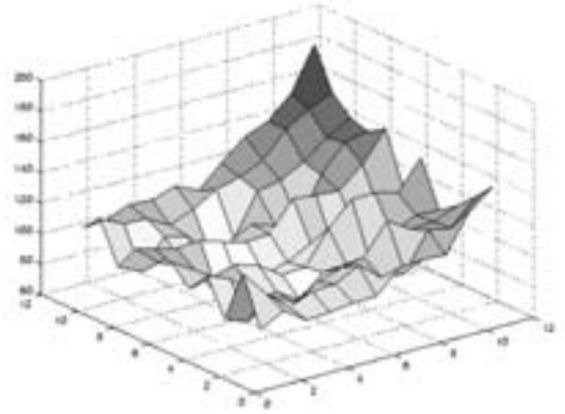
Edge: faible tous le long du contour





Sky: faible partout

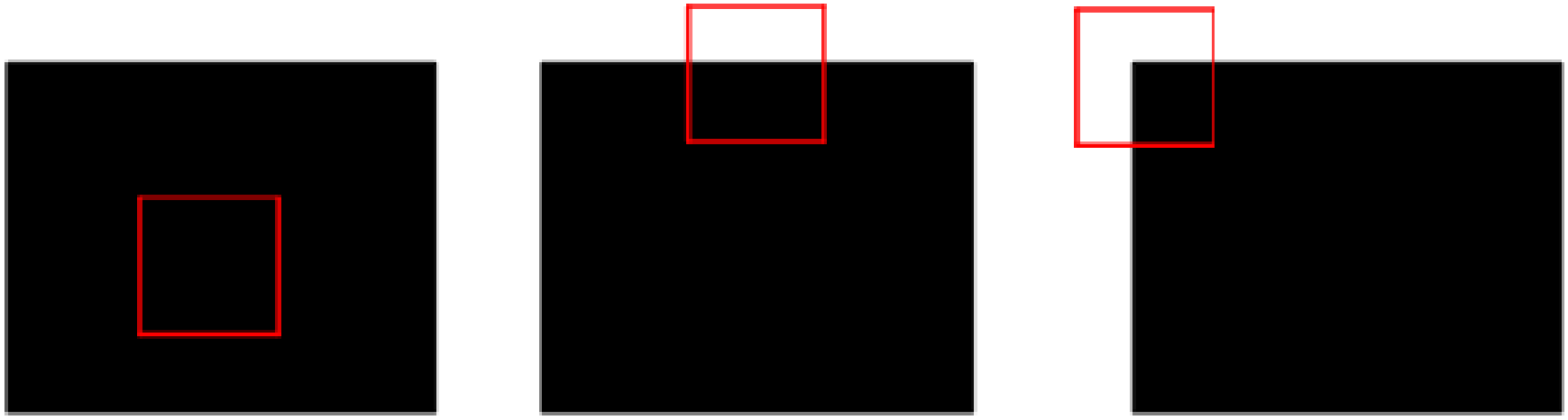




Première approche : Détecteur de Moravec

Comment obtenir une énergie forte sur les points d'interet (coins) ?

- Zones d'intensité presque constante
  - E(x; y) ~ 0
- Contours :
  - E(x; y) = 0 pour des déplacements le long du contour
  - E(x; y) > 0 pour des déplacements perpendiculaires au contour
- Coins :
  - E(x; y) > 0 pour tout (dx; dy) ≠ (0; 0)



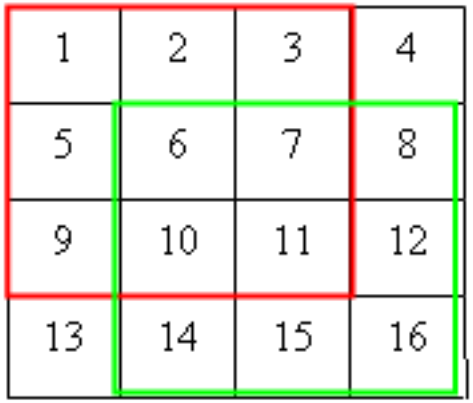
Première approche : Détecteur de Moravec

Calcul des valeurs d'énergie pour chaque direction

- $E_{1,1}(x,y)$ = somme des carrés des différences des couples

- Pour le pixel p = 6 et pour un déplacement dx = 1; dy = 1 :
  - E = Somme des carrés de

    (1-6); (2-7); (3-8); (5-10); (6-11); (7-12); (9-14); (10-15); (11-16)



Détecteur de Moravec : Algorithme

1. Pour chaque pixel (x; y) d'une image, calculer 8 valeurs de E différentes correspondant à 8 déplacements en x et y :
   - $E_1(x; y) = E_{1;0}(x; y)$
   - $E_2(x; y) = E_{1;1}(x; y)$
   - …
   - $E_8(x; y) = E_{-1;-1}(x; y)$
2. Pour chaque pixel on calcule la valeur minimum des E_i
3. On garde comme point d'intérêt les points pour lesquels ce minimum est supérieur à un seuil.

Selon le seuil on aura plus ou moins de points d'intérêts

Detecteur de Harris : Approximation des self-differences

- Calcul des self-differences ➔ toujours trop couteux
  - Pour chaque pixel : $\Sigma_{\mathbf{d}}\Sigma_{x,y}\,(I(x,y) - I(x+\mathbf{d}_x,y+\mathbf{d}_y))^2$
- Approximation
  - Explotation des gradients directionnels locaux : I_x and I_y
  - Si tous les gradients sont proches de 0
    - RAS ➔ Self-difference considérée faible
  - Si présence de gradient dans 1 direction
    - Contour → Self-difference considérée faible
  - Si présence de gradients dans plusieurs directions
    - Coin → Self-difference considérée forte
  - Mode de calcul des gradients directionnels locaux ?

Approximation de la self-difference

- Mode de calcul des gradients directionnels locaux ?
  - Exploitation de formulations matricielles
  - Sommes pondérées des gradients locaux

$$M = \quad S_w[p] = \begin{bmatrix} \sum_r w[r](I_x[p-r])^2 & \sum_r w[r]I_x[p-r]I_y[p-r] \\ \sum_r w[r]I_x[p-r]I_y[p-r] & \sum_r w[r](I_y[p-r])^2 \end{bmatrix}$$

  - Recherche de valeurs et vecteurs propres!
  - Les valeurs propres permettent de résumer la distribution des gradients locaus
    - λ_1 et λ_2 petites: Pas de gradient
    - $\lambda_1 >> \lambda_2$: gradient dans 1 seule direction
    - λ_1 et λ_2 similaire et elevées : multiple gradients → corner

Détecteur de Harris (+ en détails)

- On considère le développement limité de Taylor de la fonction d'intensité au voisinage de u,v :

$$I(x+u, y+v) = I(u,v) + x\frac{\delta I}{\delta x} + y\frac{\delta I}{\delta y} + o(x^2, y^2)$$

- Qui permet d'écrire E(x,y) sous la forme :

$$E(x,y) = \sum_{u,v} w(u,v) \left[x\frac{\delta I}{\delta x} + y\frac{\delta I}{\delta y} + o(x^2, y^2) \right]^2$$

- En négligeant le reste : $o(x^2,y^2)$ (valide pour les petits déplacements) :

$$E(x,y) = Ax^2 + 2Cxy + By^2$$

avec

W = fenêtre gaussienne

$$A = \left|\frac{\delta I}{\delta x}\right|^2 \otimes w \quad ; \quad B = \left|\frac{\delta I}{\delta y}\right|^2 \otimes w \quad ; \quad C = \left(\frac{\delta I}{\delta x}\frac{\delta I}{\delta y}\right) \otimes w$$

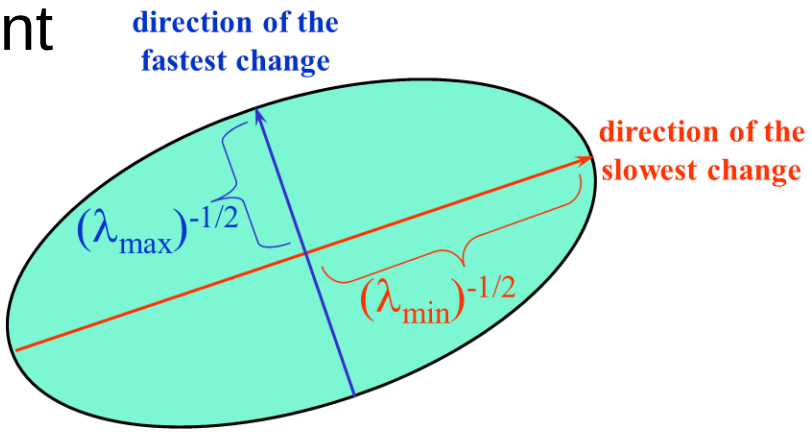
Détecteur de Harris (+ en détails)

- Ecriture sous forme matricielle :

$$E(x,y) = (x,y) \cdot M \cdot (x,y)^t$$

$$A = \frac{\delta I}{\delta x}^2 \otimes w \quad ; \quad B = \frac{\delta I}{\delta y}^2 \otimes w \quad ; \quad C = (\frac{\delta I}{\delta x}\frac{\delta I}{\delta y}) \otimes w$$

- M est symétrique et définie positive ➔ décomposition en valeurs propres
- Les valeurs propres de M correspondent aux courbures principales associées à *E*



Detecteur de Harris (+ en détails)

Plutôt que de calculer les valeurs propres, il est possible de calculer :

$$\det(M) = AB - C^2 = \lambda_1.\lambda_2$$

$$\text{trace}(M) = A + B = \lambda_1 + \lambda_2$$

Et la réponse :

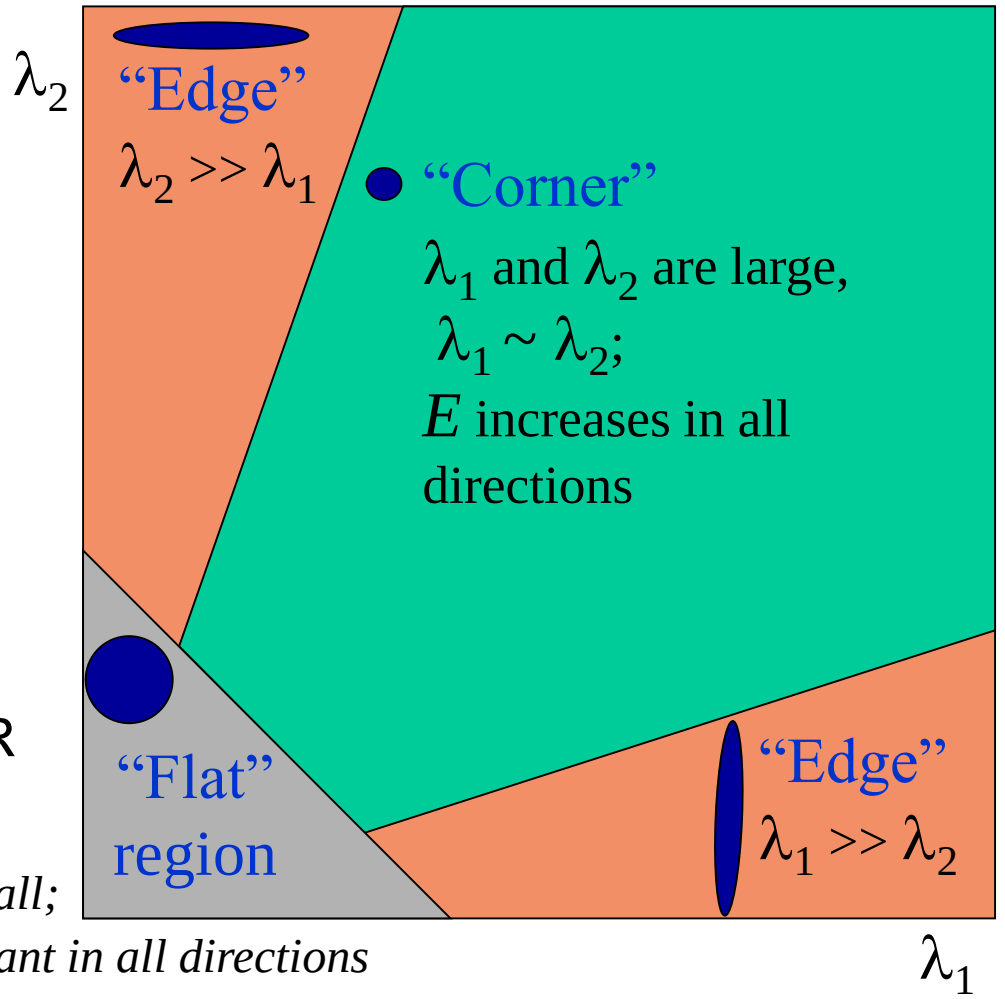
$$R = \det(M) - k\ \text{trace}^2(M)$$

k = 0.04

Les valeurs de R sont

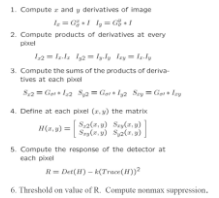
- positives au voisinage d'un coin,
- négatives au voisinage d'un contour
- faibles dans une région d'intensite cstte

⇒ coins/point d'intéret = max locaux de R



λ_1 and λ_2 are small;
E is almost constant in all directions

Détecteur de Harris : Algorithme



//Calcul des dérivées de l'image en ligne et en colonne (gradient):
Ix (x , y) = ∂ (x , y) I ∂x
Iy (x , y) = ∂(x , y) I ∂y

//Calcul des matrices, dérivées aux carrés + leur multiplication
$Ix^2(x,y)$
$Iy^2(x,y)$
Ixy(x,y) = Ix(x,y) · Iy(x,y)

//Calcul des somme pondérée des dérivées = Lissage Gaussien des matrices
K(x,y) = K(x,y) * Hgauss
//avec K = Ix^2, Iy^2, Ixy et Hgauss le filtre gaussien.

//Calcul de l'Energie en chaque pixel
Calcul de M(x,y)

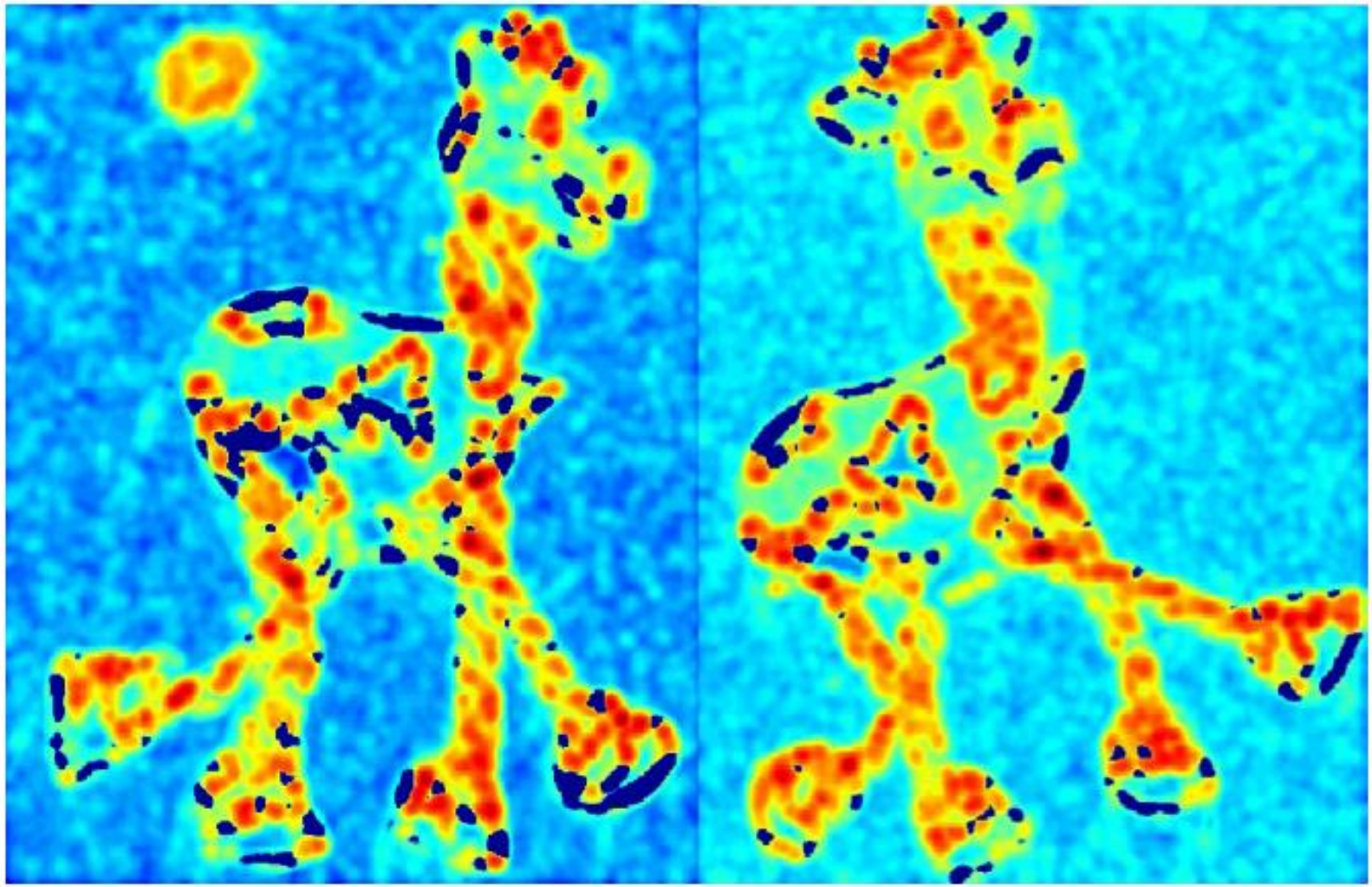
$$M = \begin{bmatrix} A & C \\ C & B \end{bmatrix}$$

$R(x , y) = A.B - C^2 - \alpha\cdot(A + B)^2$

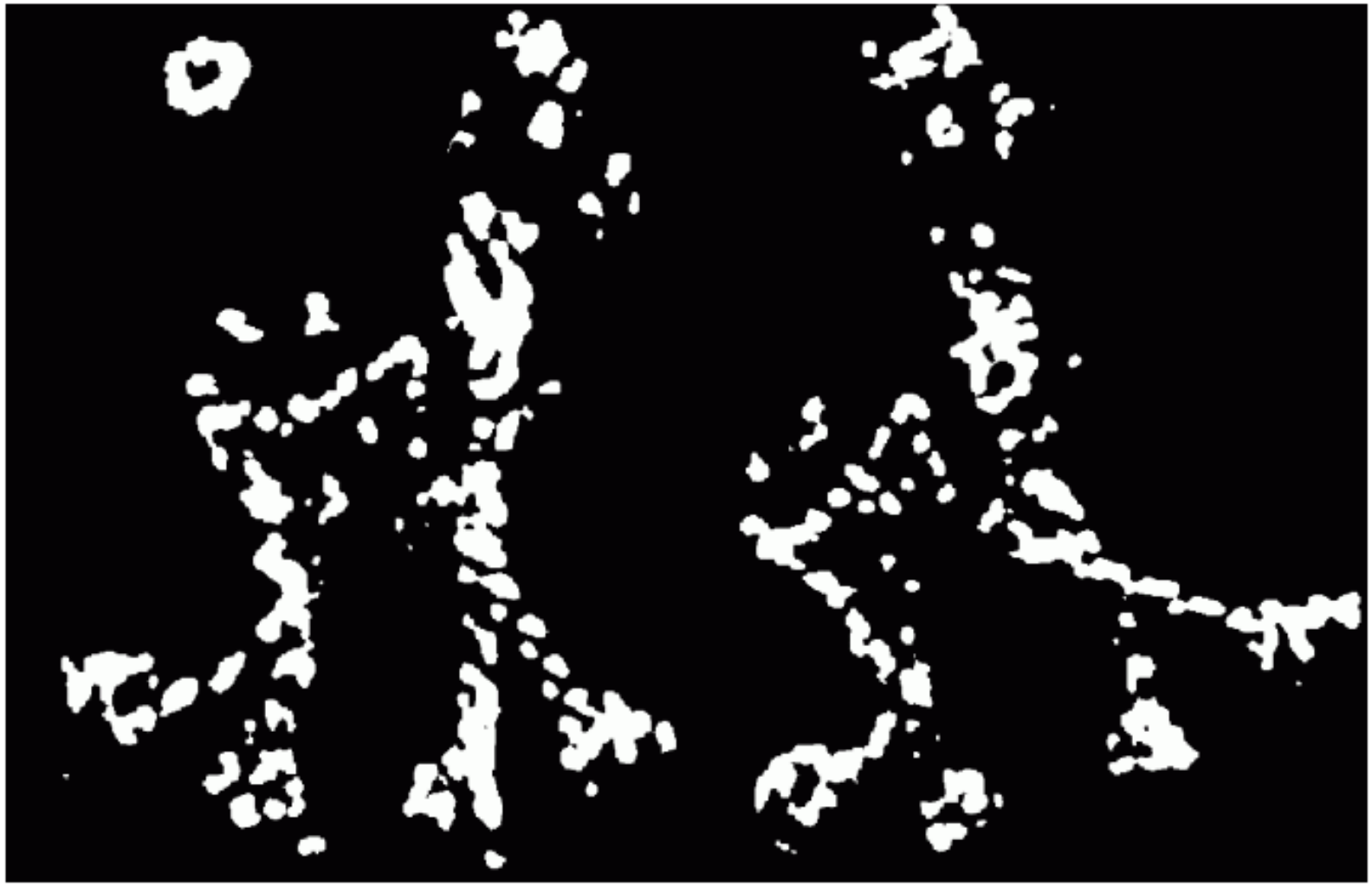
α est la sensibilité du détecteur.
R est appelée "corner response function" CRF
Plus α est grand plus la sensibilité est faible et moins le nombre de coin sera grand.
Typiquement α est compris entre 0.04 et 0.06 mais ne dépasse pas 0,25.

//Seuillage pour éliminer les faibles valeurs du CRF
R(x , y) < T = 0
//Généralement ce seuil (T) est entre 104 et 107

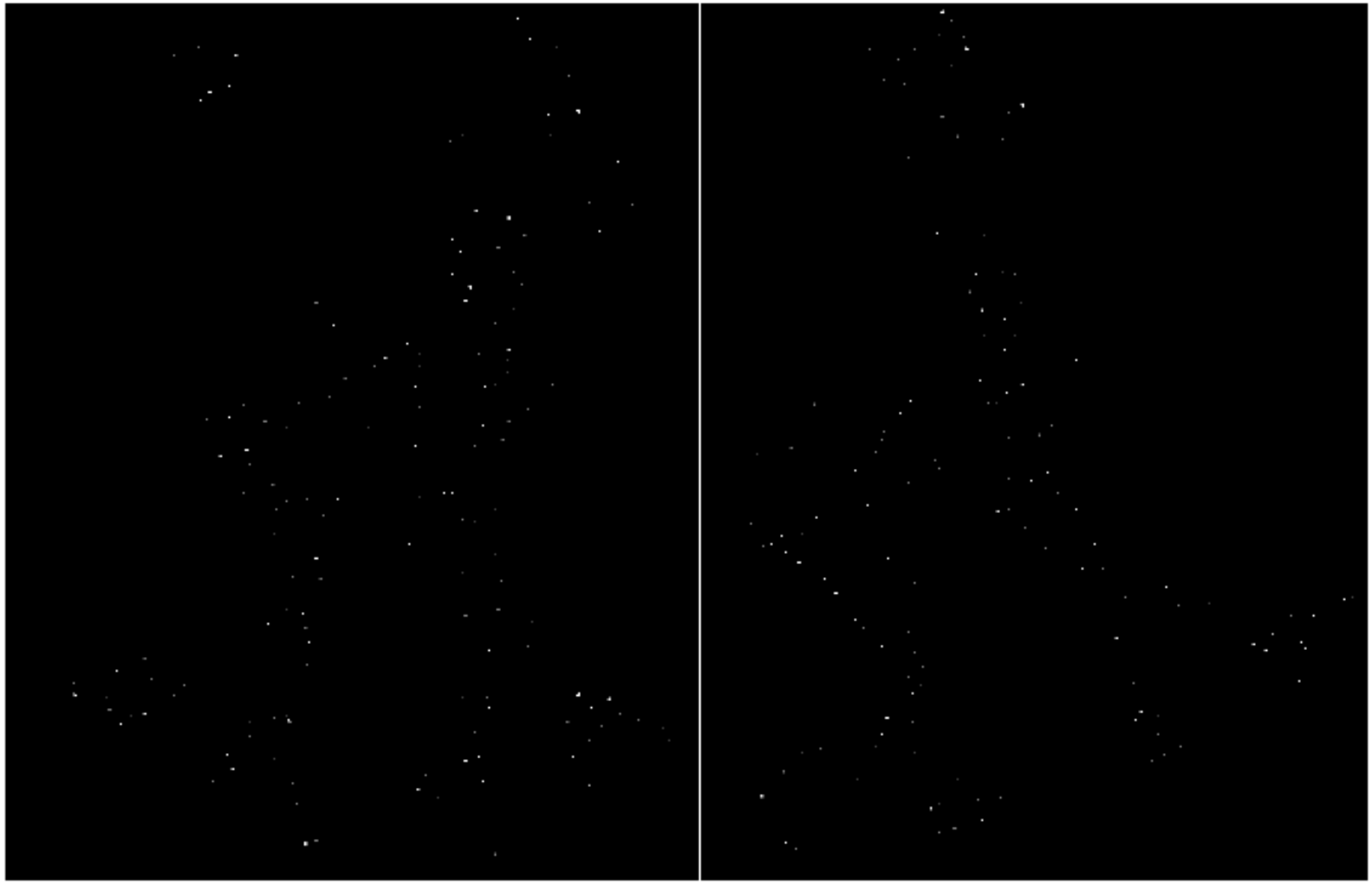
//Stockage des coins détectés par ordre décroissant
//Détection de maxima locaux
//(pour éliminer les coins trop proches les uns des autres)



Visualisation de la réponse : R



Seuillage sur R



Maxima locaux.

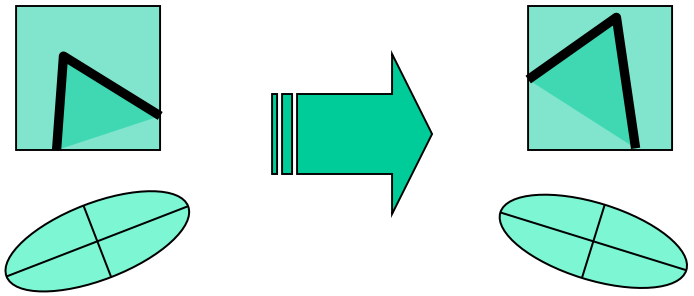
Detecteur de Harris : Choix du seuil ?

R_{min}=100

R_{min}= 150

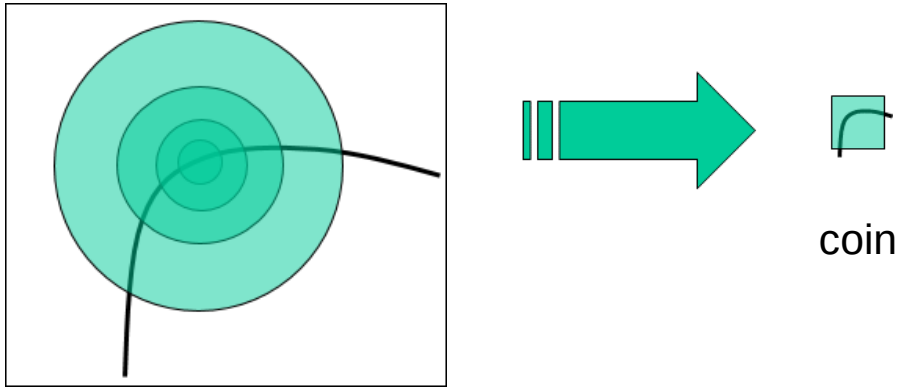
Harris : Propriétés

- Invariance à la rotation



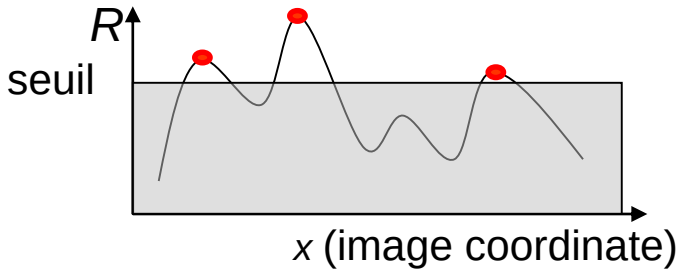
Les valeurs propres restent identiques lorsque la forme change d'orientation

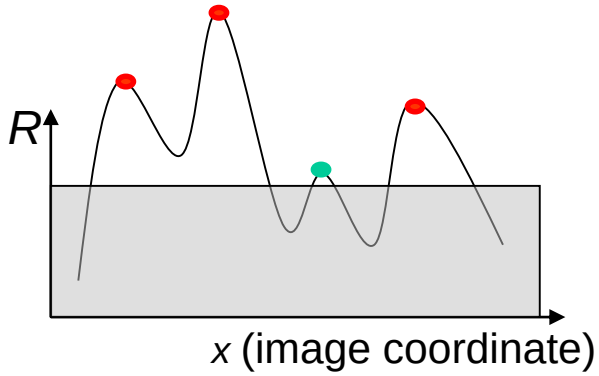
- MAIS pas invariant au changement d'echelle



- Invariance partielle aux variations d'intensité

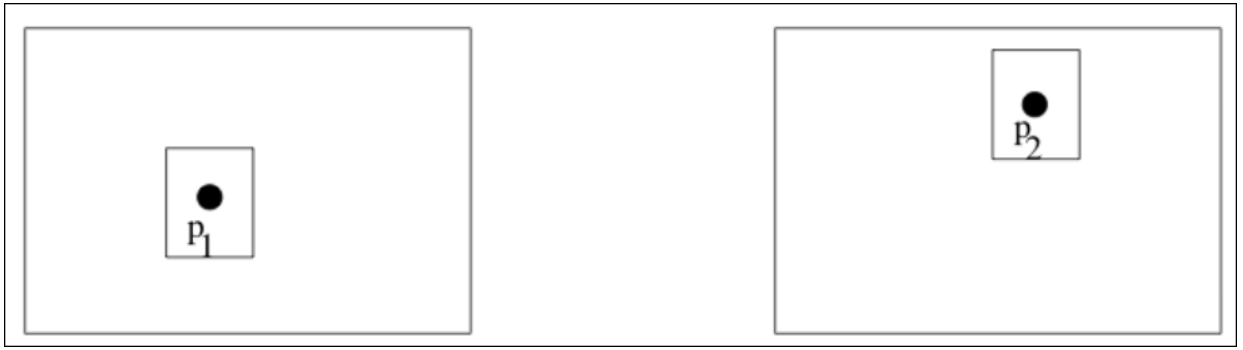
Calcul de dérivées ➔ invariance

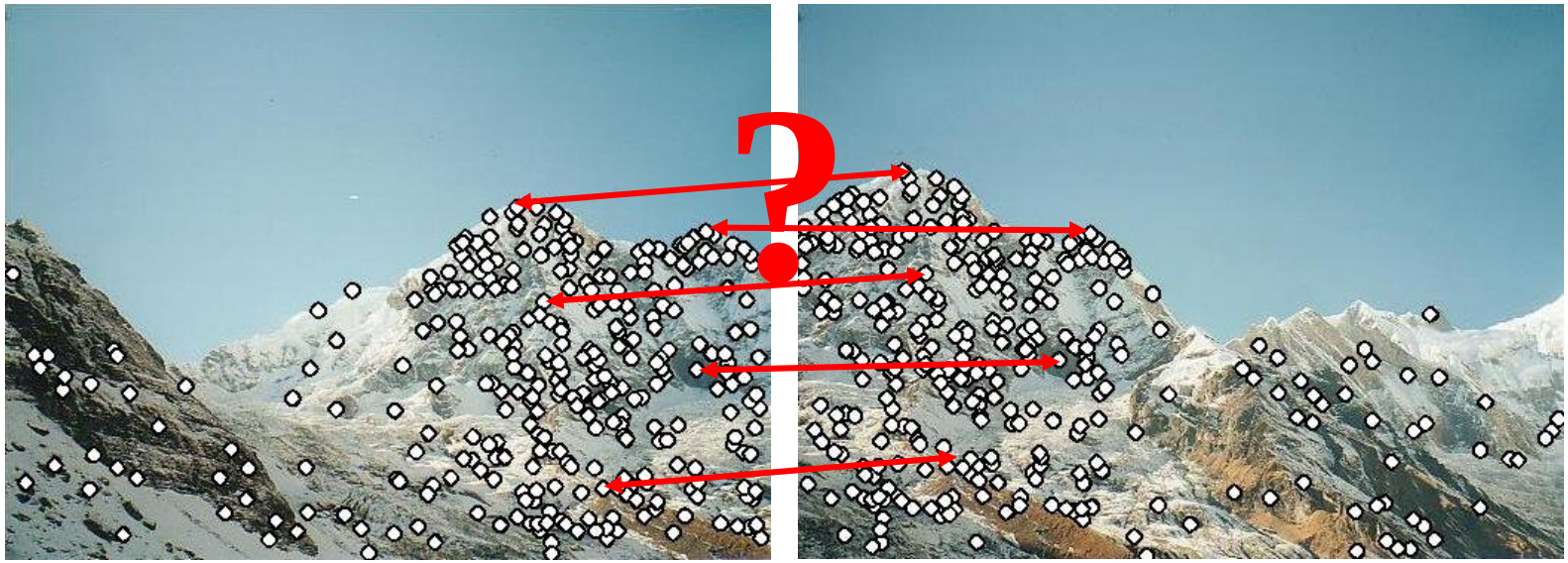




Mise en correspondance : Descripteurs de points

- On sait detecter des points d'intéret
- Question suivante : **Mise en correspondance ?**

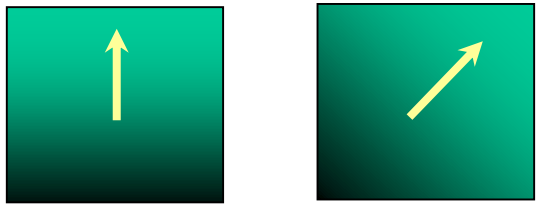




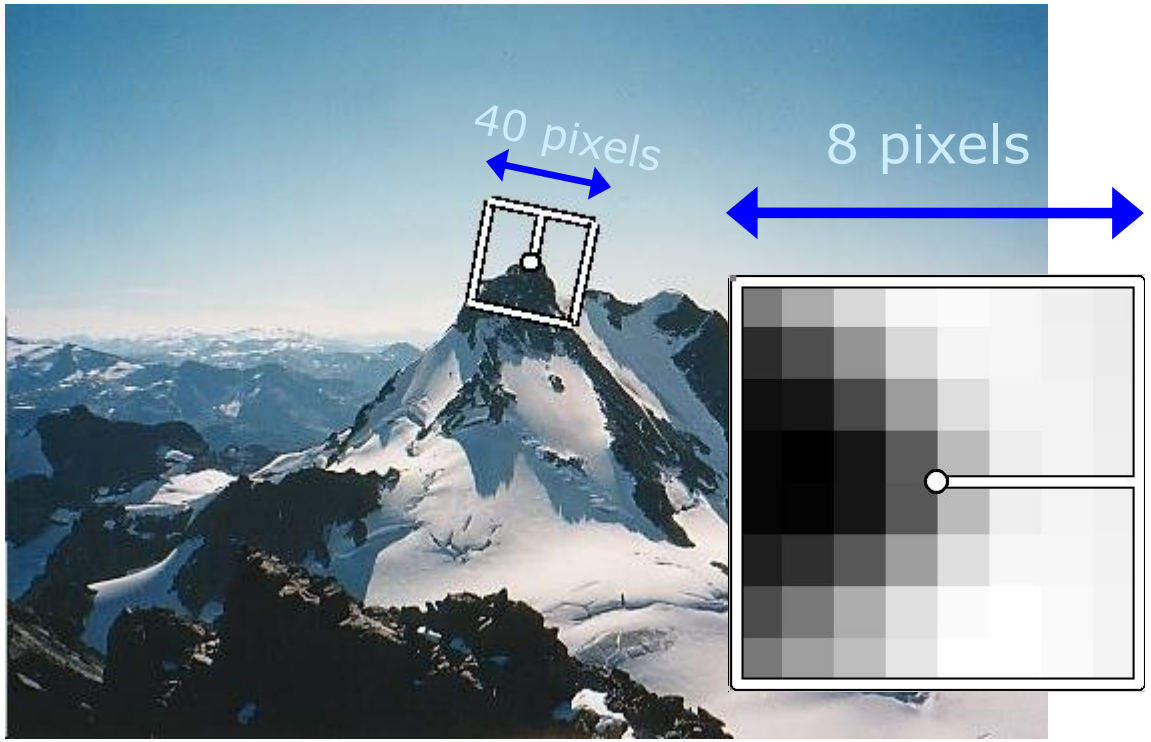
Descripteurs de points : MOPS

[Brown, Szeliski, Winder, CVPR'2005]

- **Descripteur = un vecteur $\in \mathfrak{R}^n$**
- Assurer l'invariance à la rotation
  - Recherche de l'orientation dominante localement (du gradient)
  - Extraction de "patches" relativement à cette orientation



- Caractèriser
  - Patch 8x8
  - 5 échelles d'analyse
  - Normalisation: des intensités



MOPS : Analyse multi-échelles

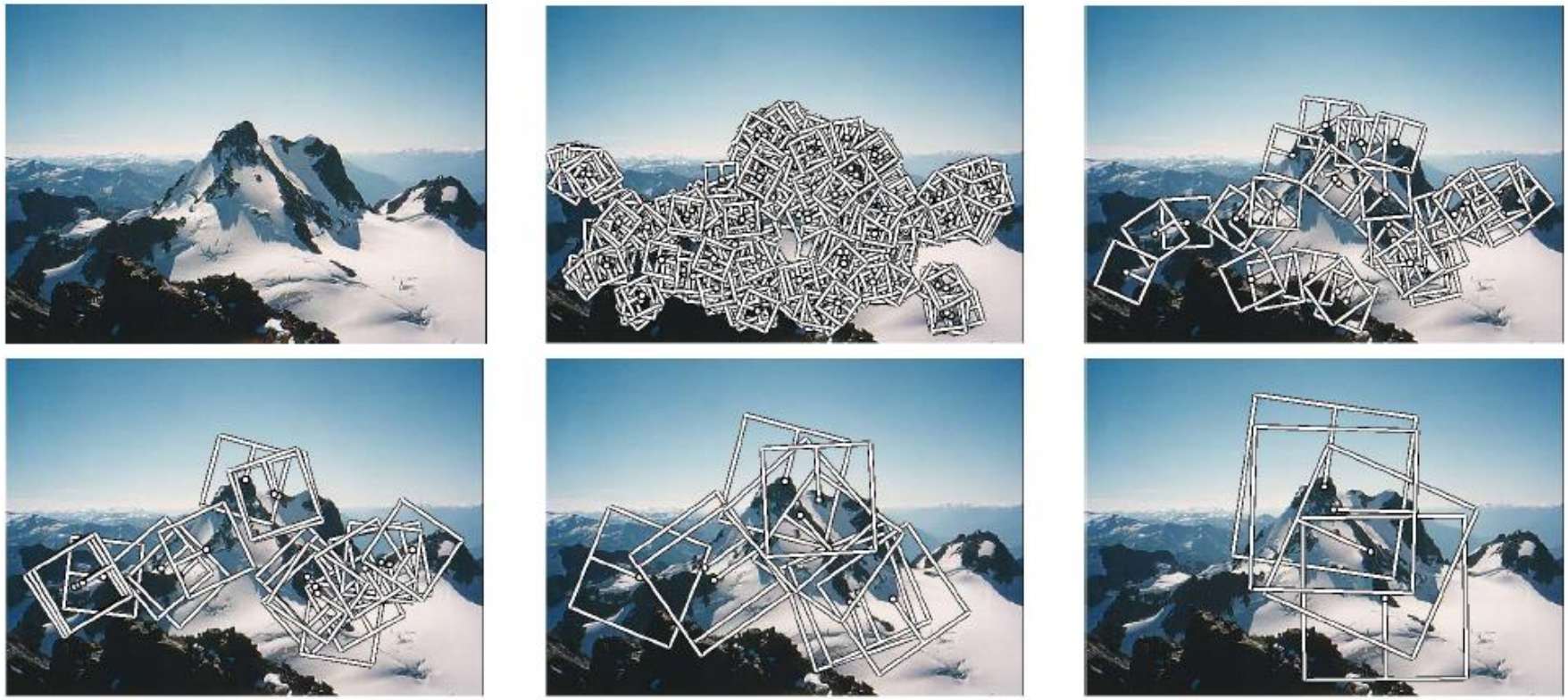
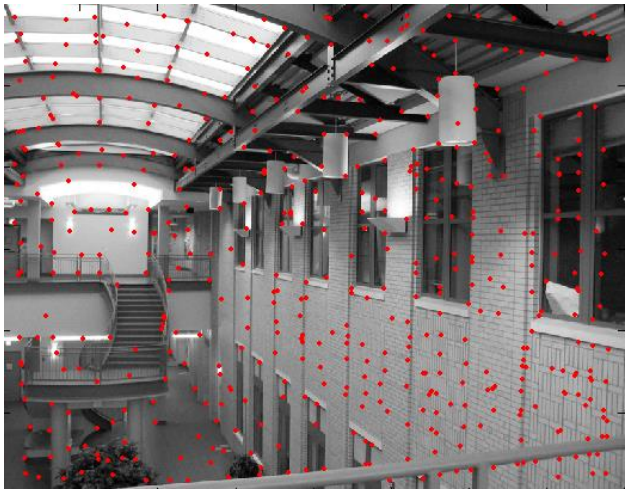


Figure 1. Multi-scale Oriented Patches (MOPS) extracted at five pyramid levels from one of the Matier images. The boxes show the feature orientation and the region from which the descriptor vector is sampled.

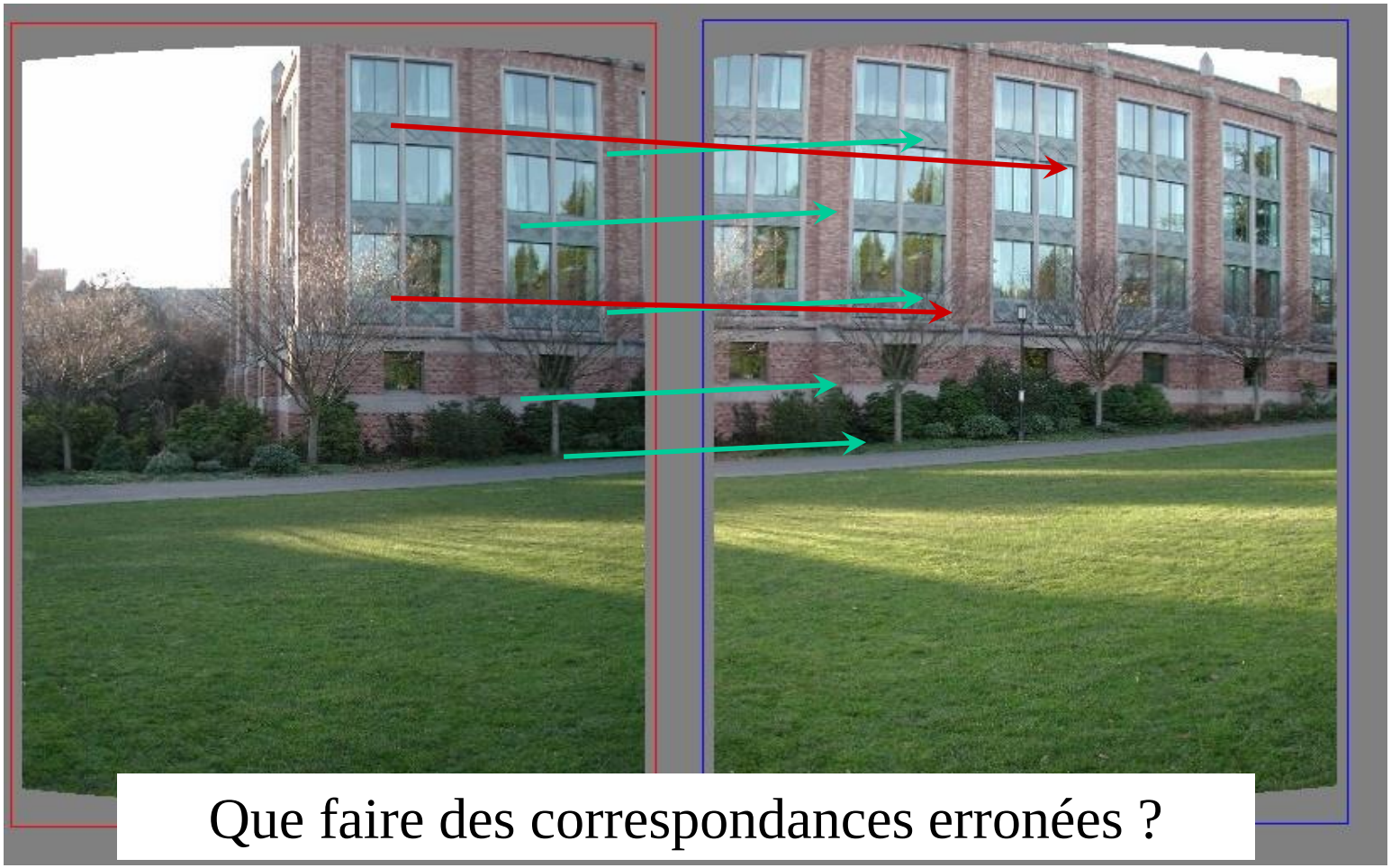
Mise en correspondance

- Recherche exhaustive des correspondances
  - Pour chaque point dans l'image 1, calcul d'une similarité (SDD) avec tous les autres points dans l(es) image(s)
- Hashing
  - Calcul d'un hash code pour chaque descripteur
- Techniques des plus proches voisins
  - *kd*-trees et ses variants

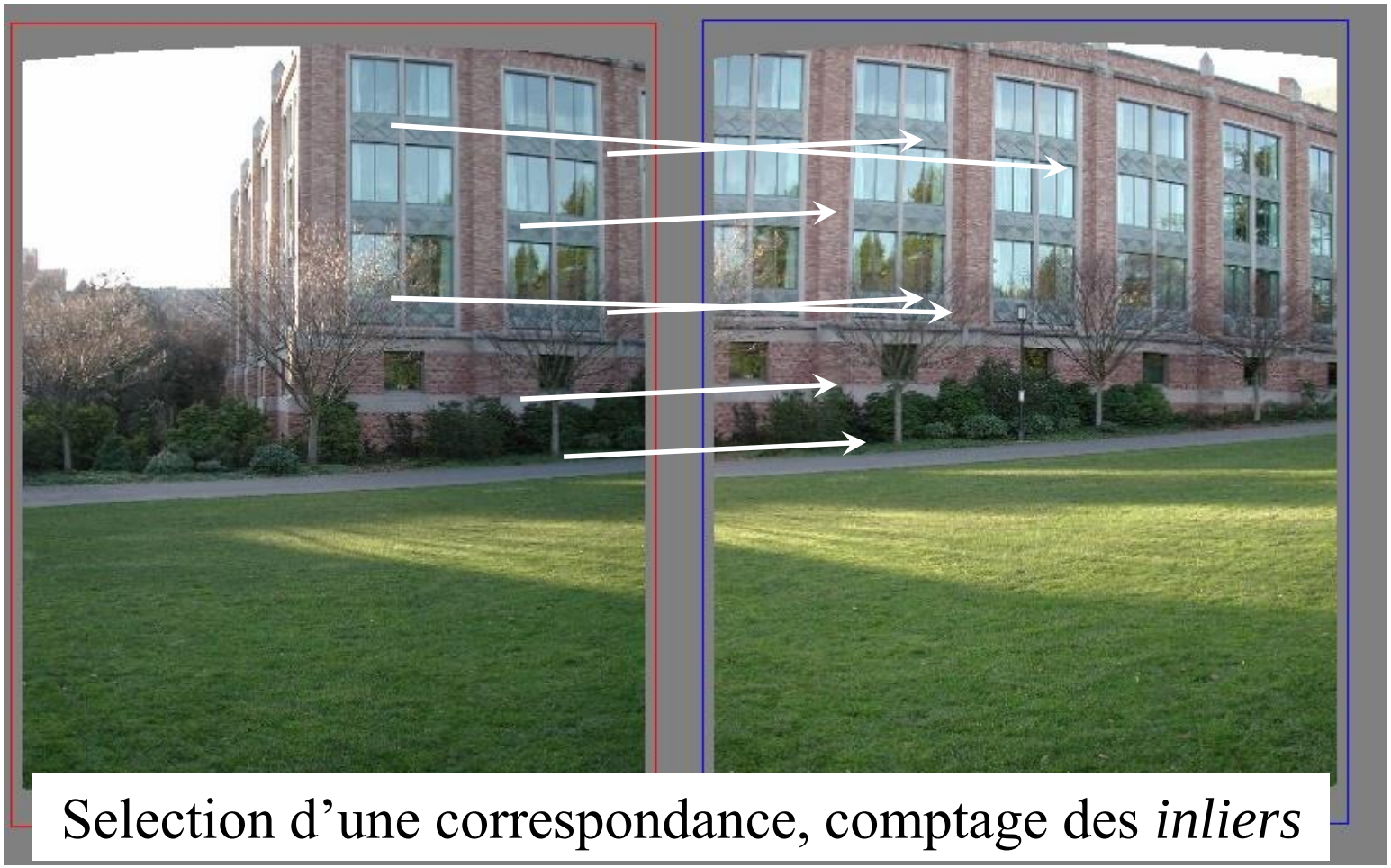
- Gestion des "outliers"
  - SSD(patch1,patch2) < Seuil
  - Choix du seuil ?



Mise en correspondance

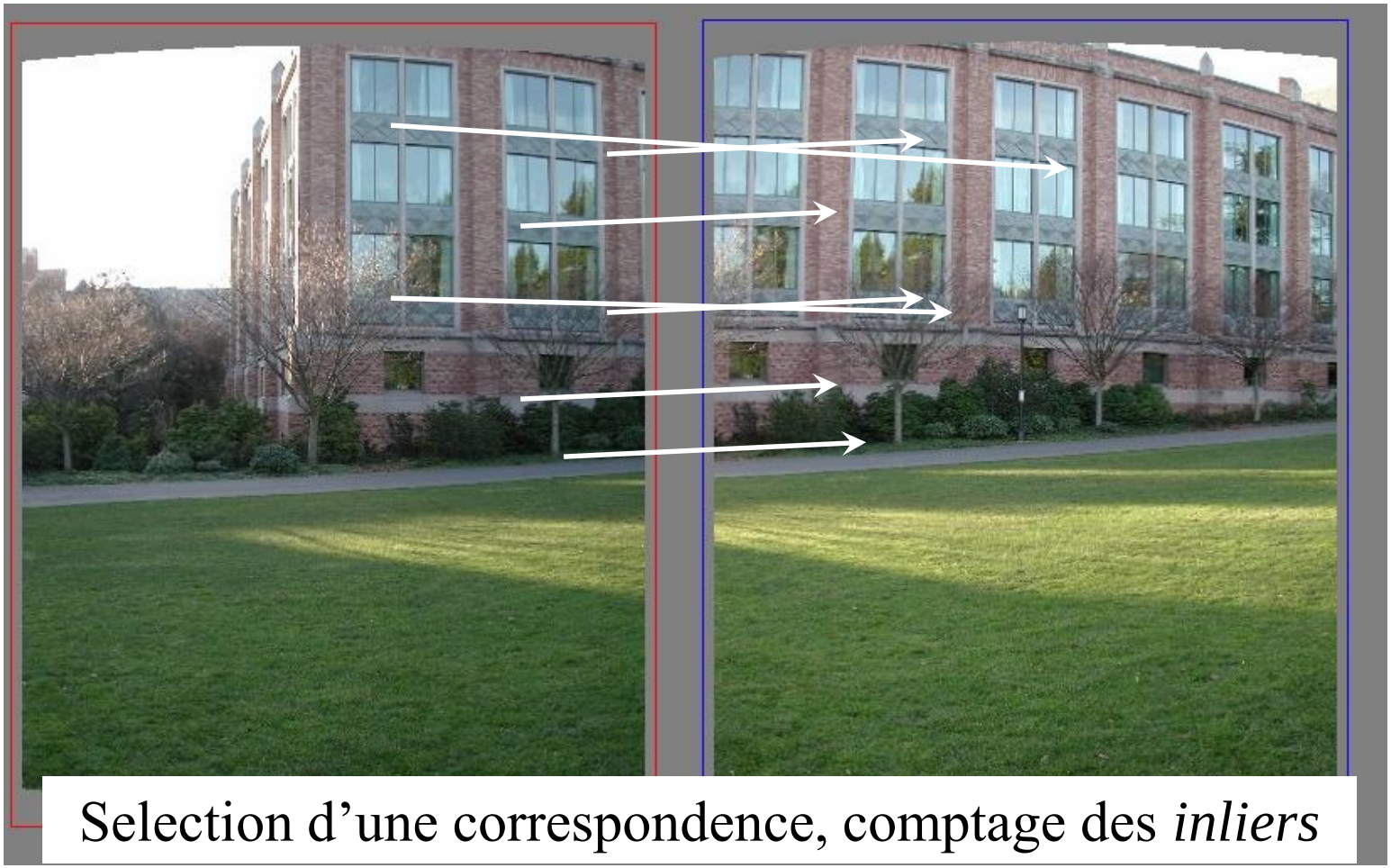


RAndom SAmple Consensus



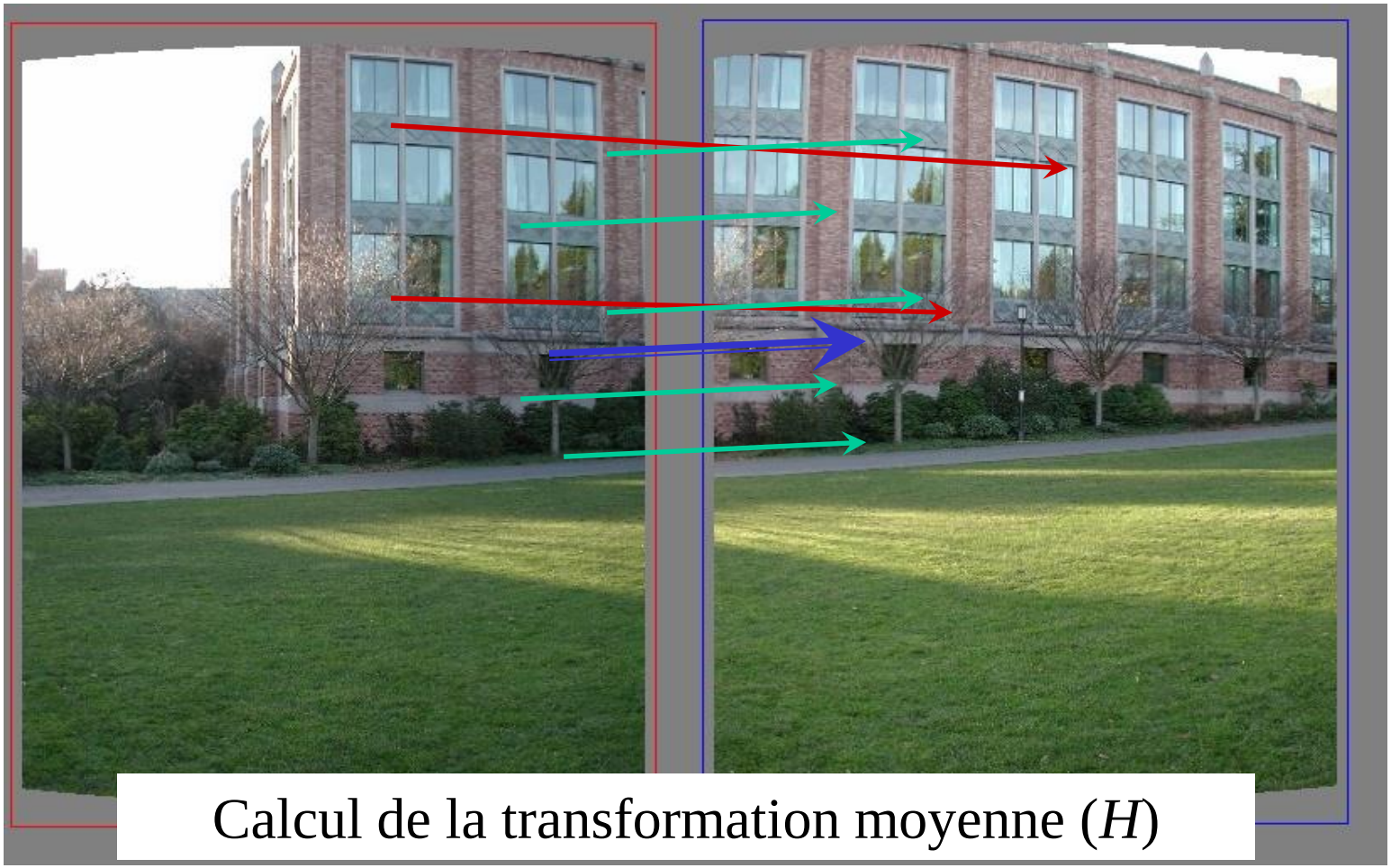
Selection d’une correspondance, comptage des *inliers*

RAndom SAmple Consensus



Selection d'une correspondence, comptage des *inliers*

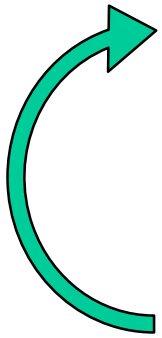
RAndom SAmple Consensus



Calcul de la transformation moyenne (*H*)

RAndom SAmple Consensus

RANSAC loop :



1. Select four feature pairs (at random)
2. Compute homography *H* (exact)
3. Compute *inliers* where $SSD(p_i', \mathbf{H}\, p_i) < \varepsilon$
4. Record the largest set of inliers so far
5. Re-compute least-squares H estimate on the largest set of the inliers

RANSAC en général

RANSAC = Random Sample Consensus

- Un algorithme pour la détection de modèle de transformation (homographie) en présence de nombreux outliers

- Etant donné N points x_i, dont on suppose que la majorité d'entre eux ont été générés par un modele de paramètres Θ, l'objectif est de retrouver Θ.

RANSAC algorithm

Combien de fois?

Combien ?
Un minimum

Faire k fois:

(1) choisir n exemples aleatoirement

(2) Adapter les parametres Θ avec ces n exemples

(3) Pour chacun des autres N-n points, calculer la distance (erreur) au modele, compter le nombre de inliers, c

Renvoyer Θ correspondant au plus grand c

c = inliers = corrects

Choix de k ?

n: nombre d'exemples choisis à chaque itération

p: probabilité de vrai inliers (risqué d'erreurs dans les couplages proposées = 0,5)

P: probabilité d'avoir au minimum 1 succès après **k** essais

$$\mathrm{P} = 1 - (1 - \underbrace{p^n}_{\text{n exemples sont tous inliers}})^{\underbrace{k}_{\text{Une erreur}}}$$

Erreur apres k essais

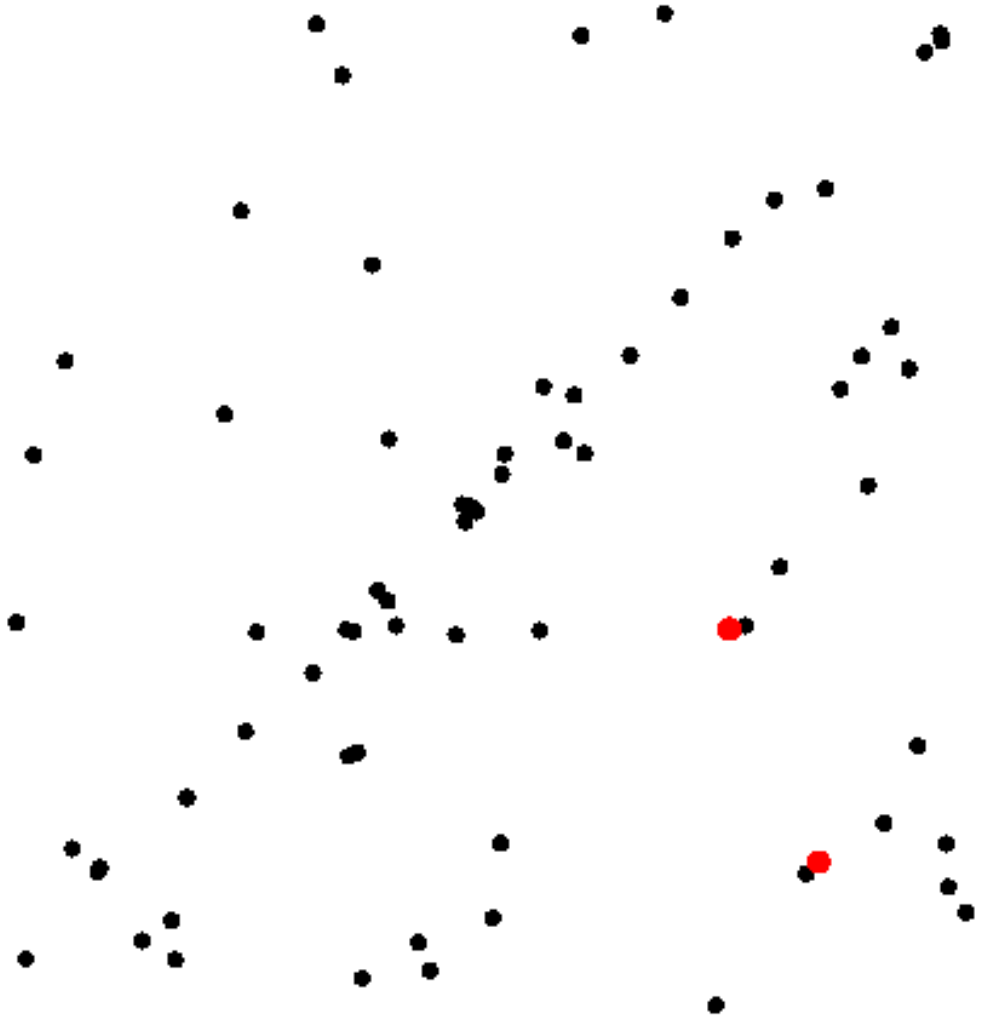
$$k = \frac{\log(1-\mathrm{P})}{\log(1-p^n)}$$

➔ Pour **P**=*0.99* ➔

n	*p*	*k*
3	0.5	35
6	0.6	97
6	0.5	293

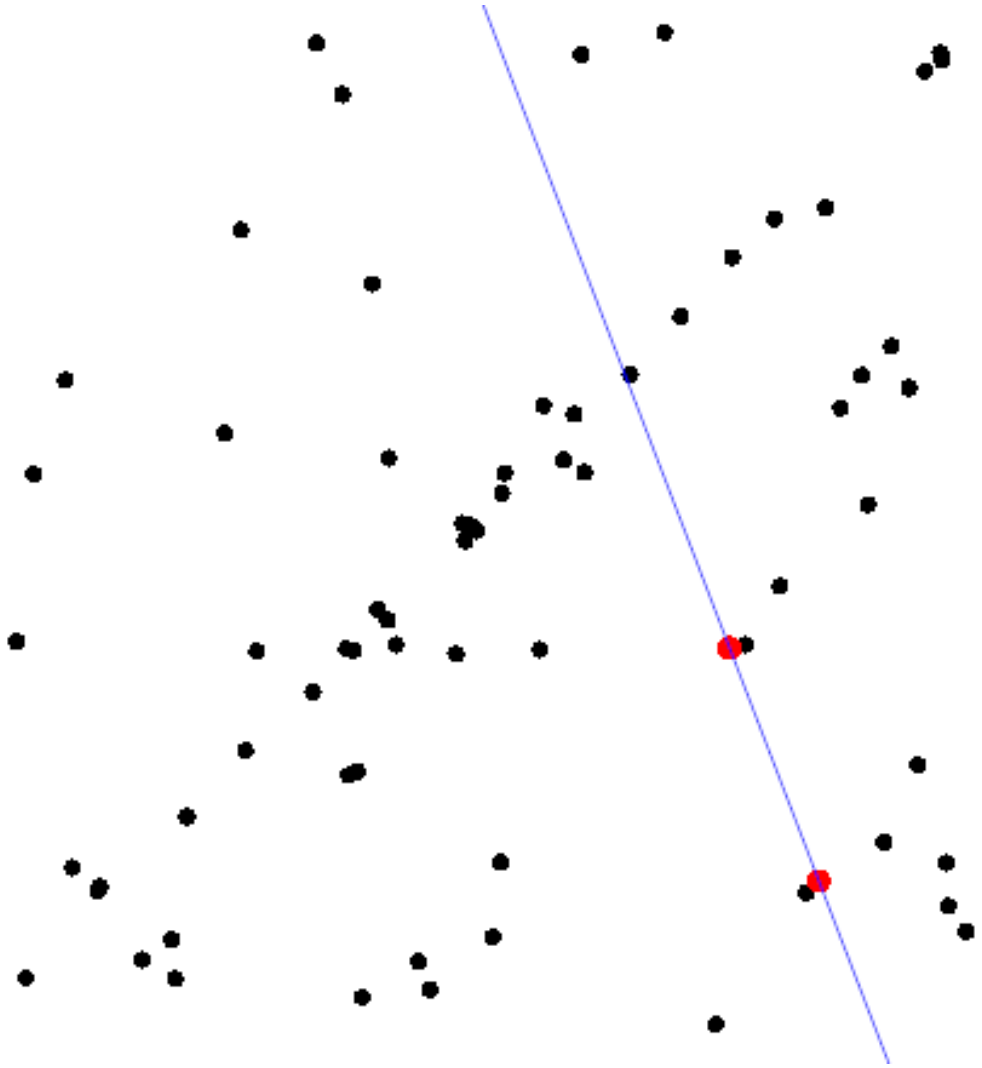
Exemple: Estimation d'une droite

Exemple: Estimation d'une droite

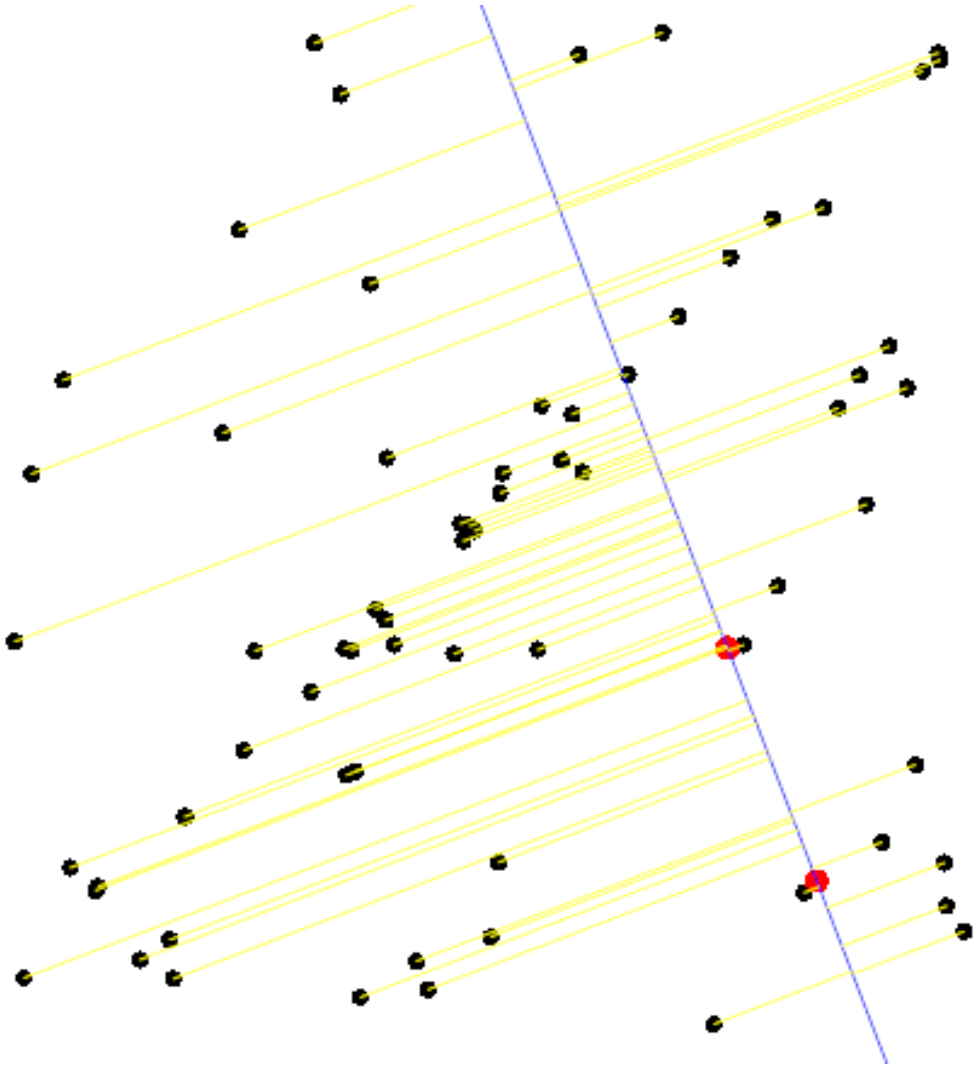


$n=2$ *(nb exemples choisis)*

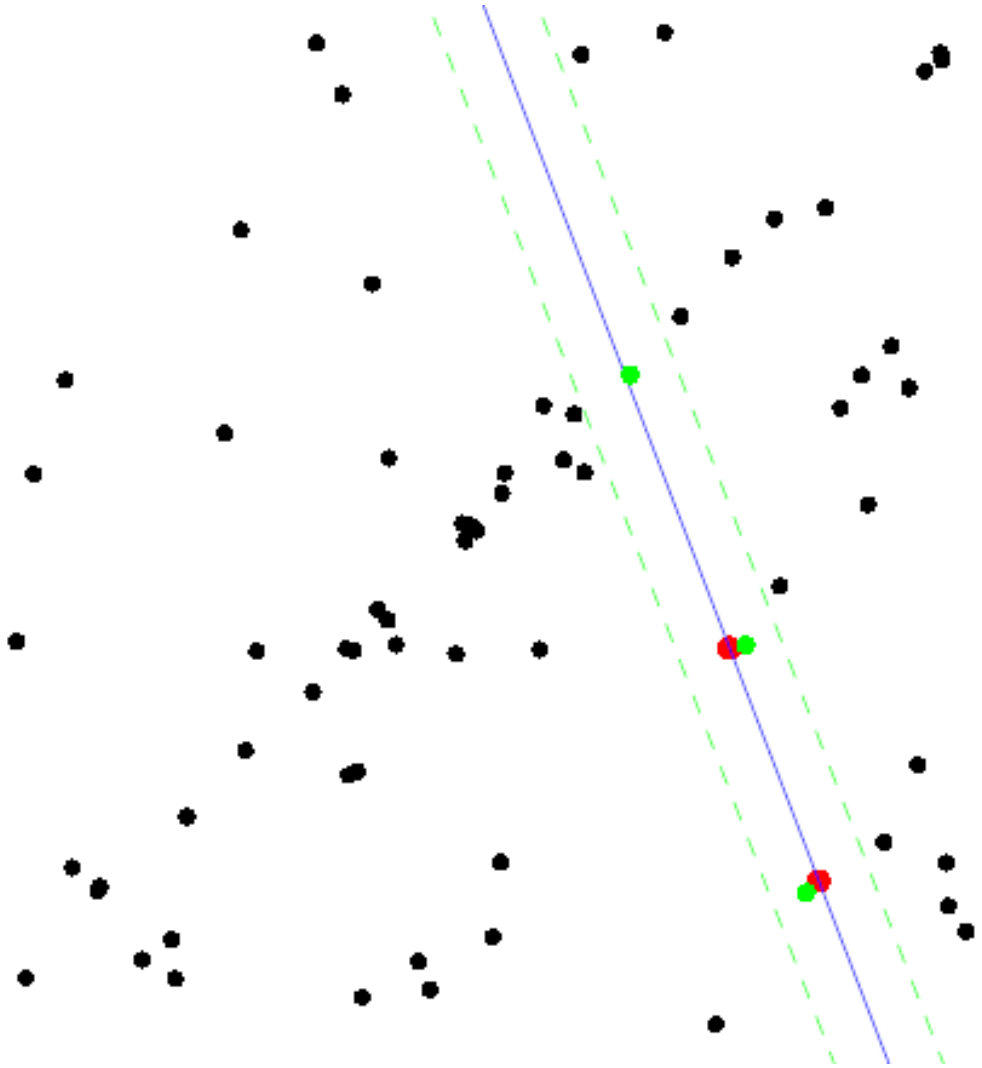
Exemple: Estimation d'une droite



Mesure des distances

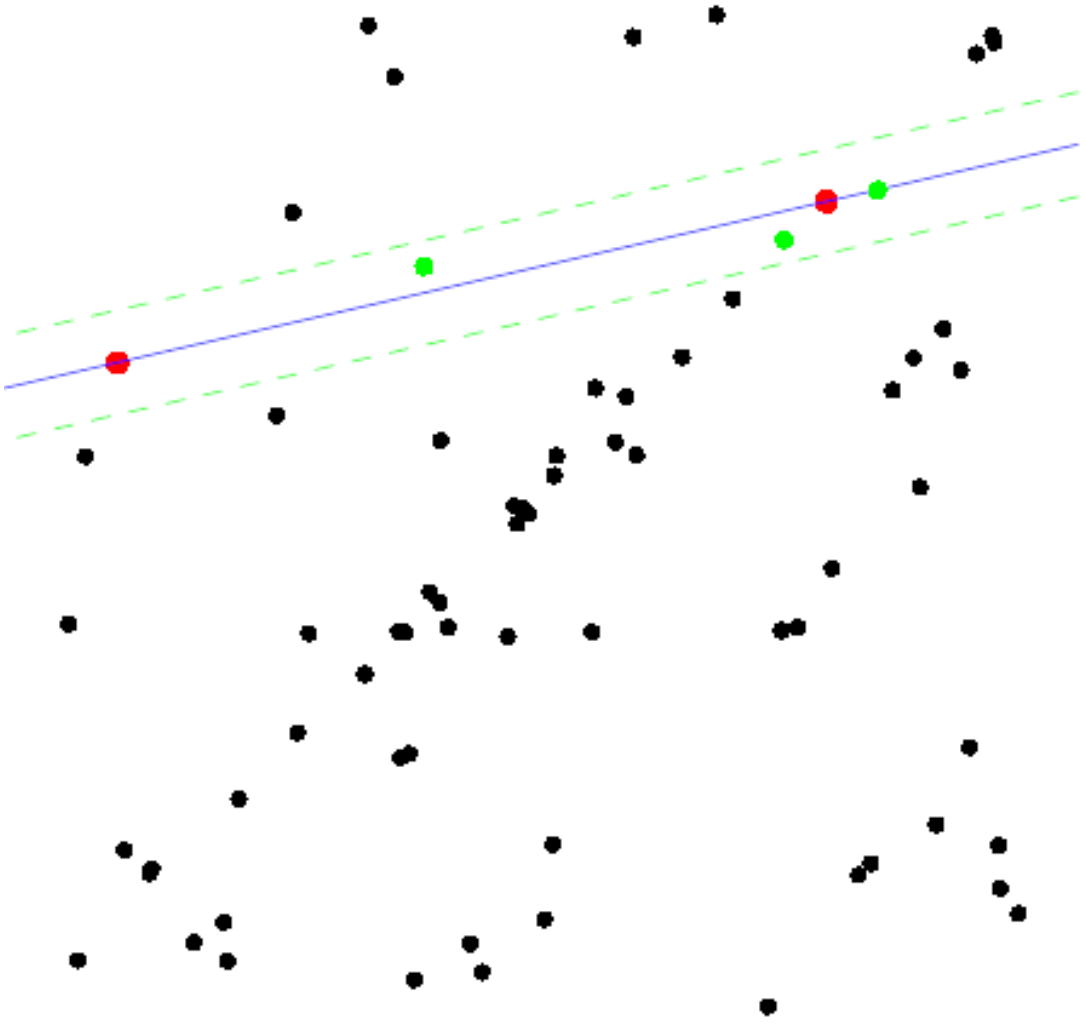


Nombre d'inliers



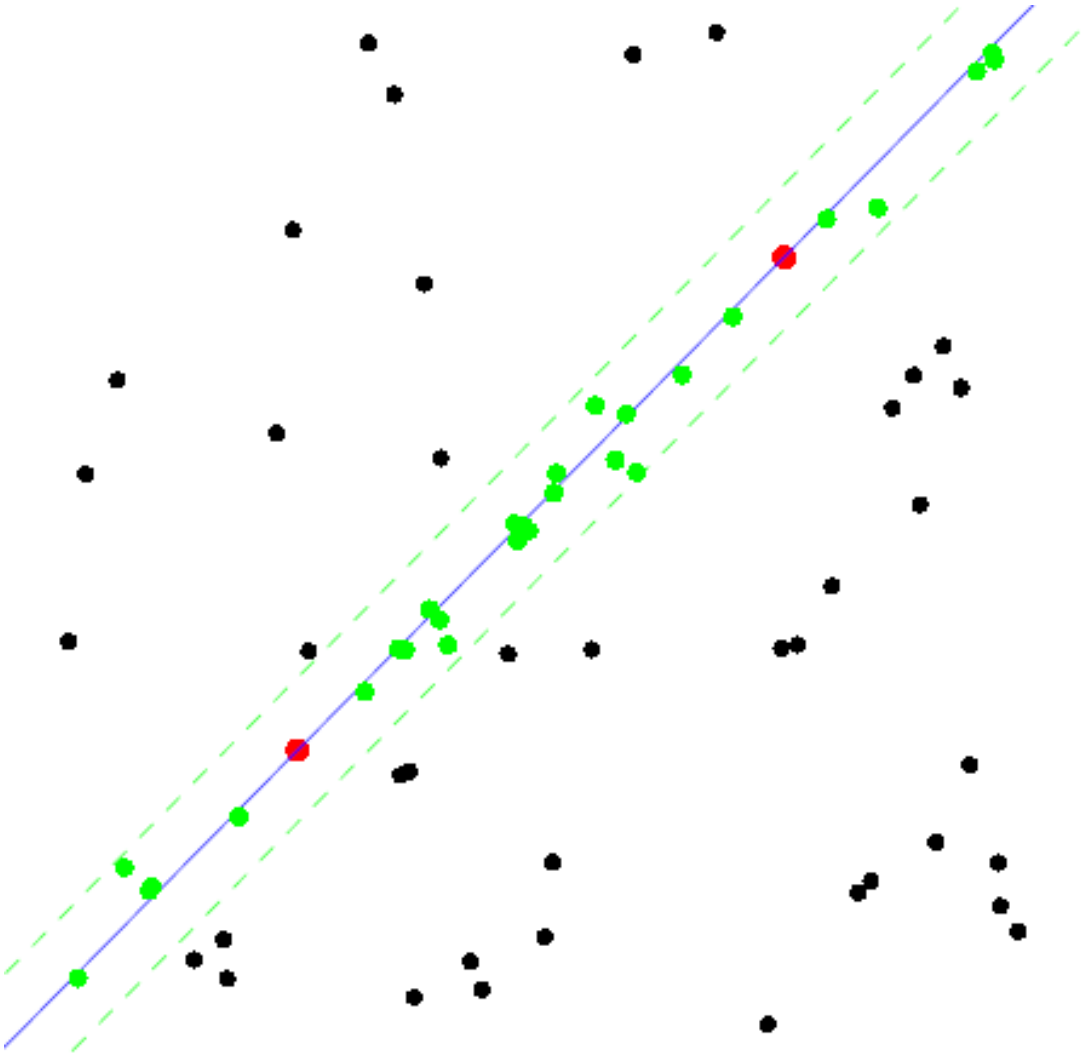
$c=3$

Autre essai



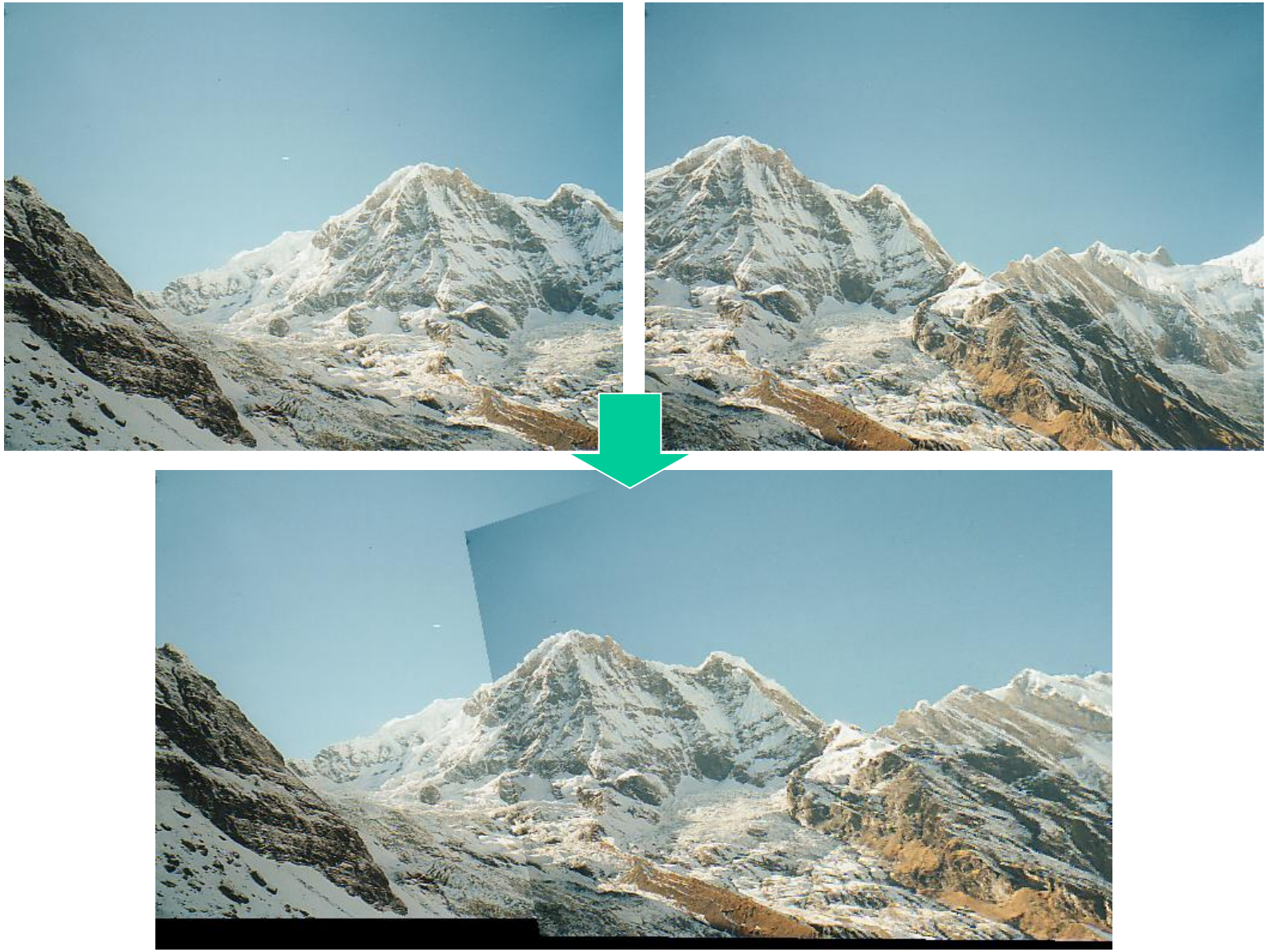
$c=3$

Le meilleur modele

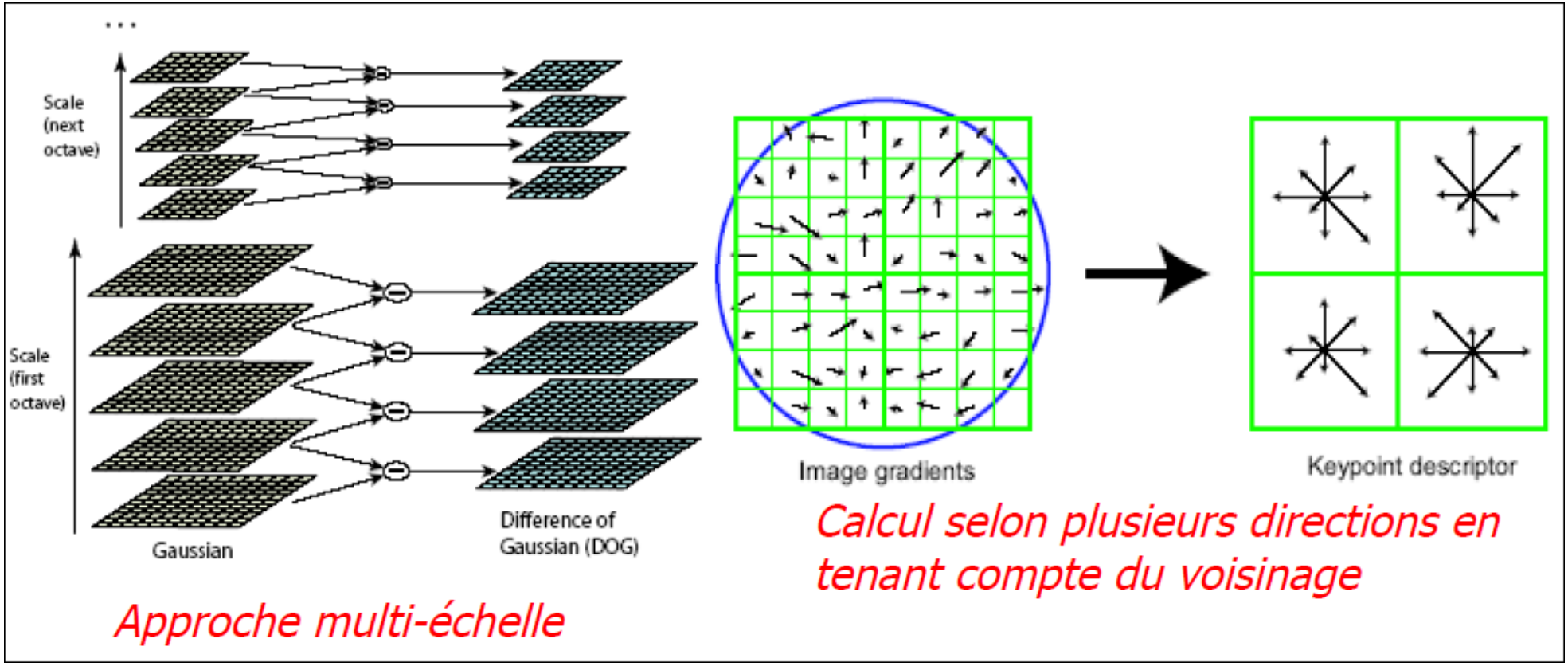


c=15

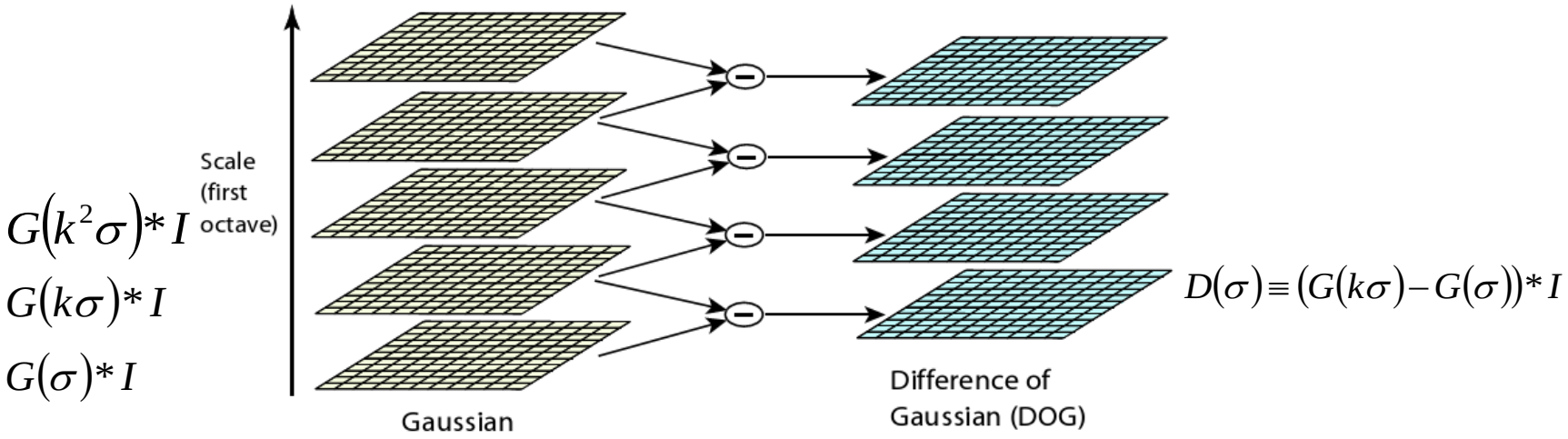
Autre application : Reconstruction de panorama

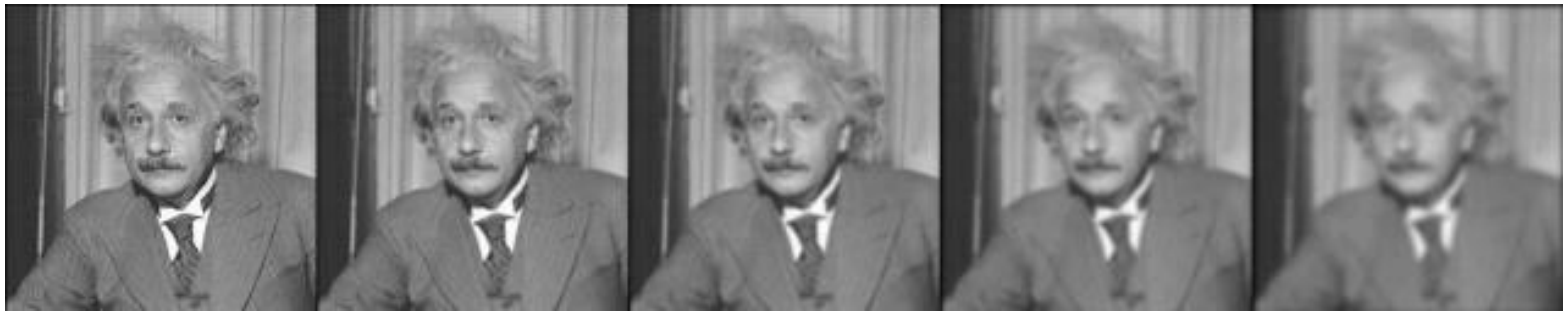


La star : SIFT


Scale
(next
octave)
Scale
(first
octave)
Gaussian
Difference of
Gaussian (DOG)
Image gradients
Keypoint descriptor
Approche multi-échelle
Calcul selon plusieurs directions en
tenant compte du voisinage

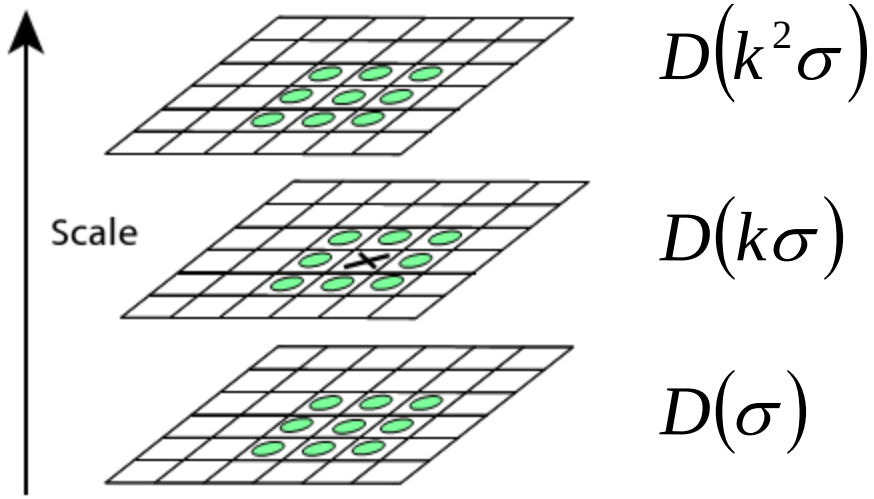
Difference-of-Gaussians





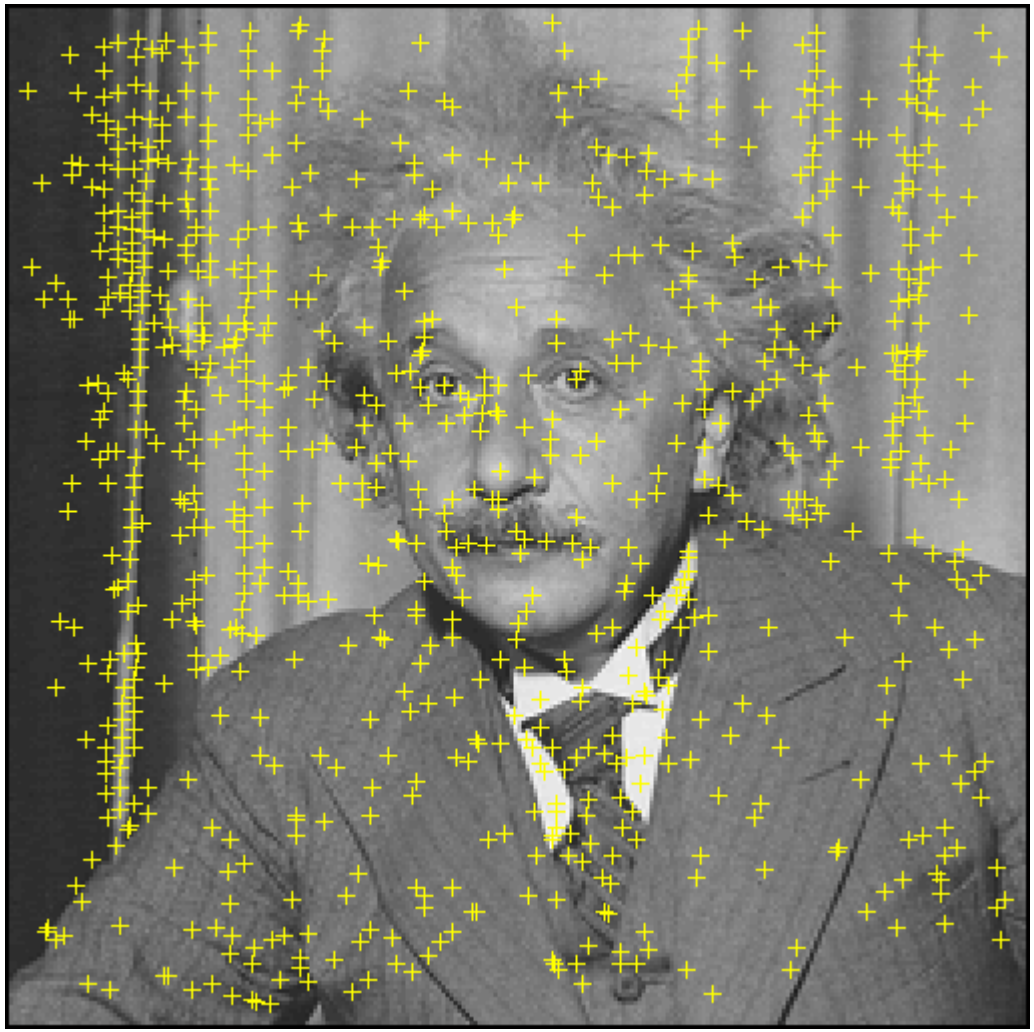
Scale-Space Extrema

- Conservation de tous les extrema dans un voisinage 3x3x3

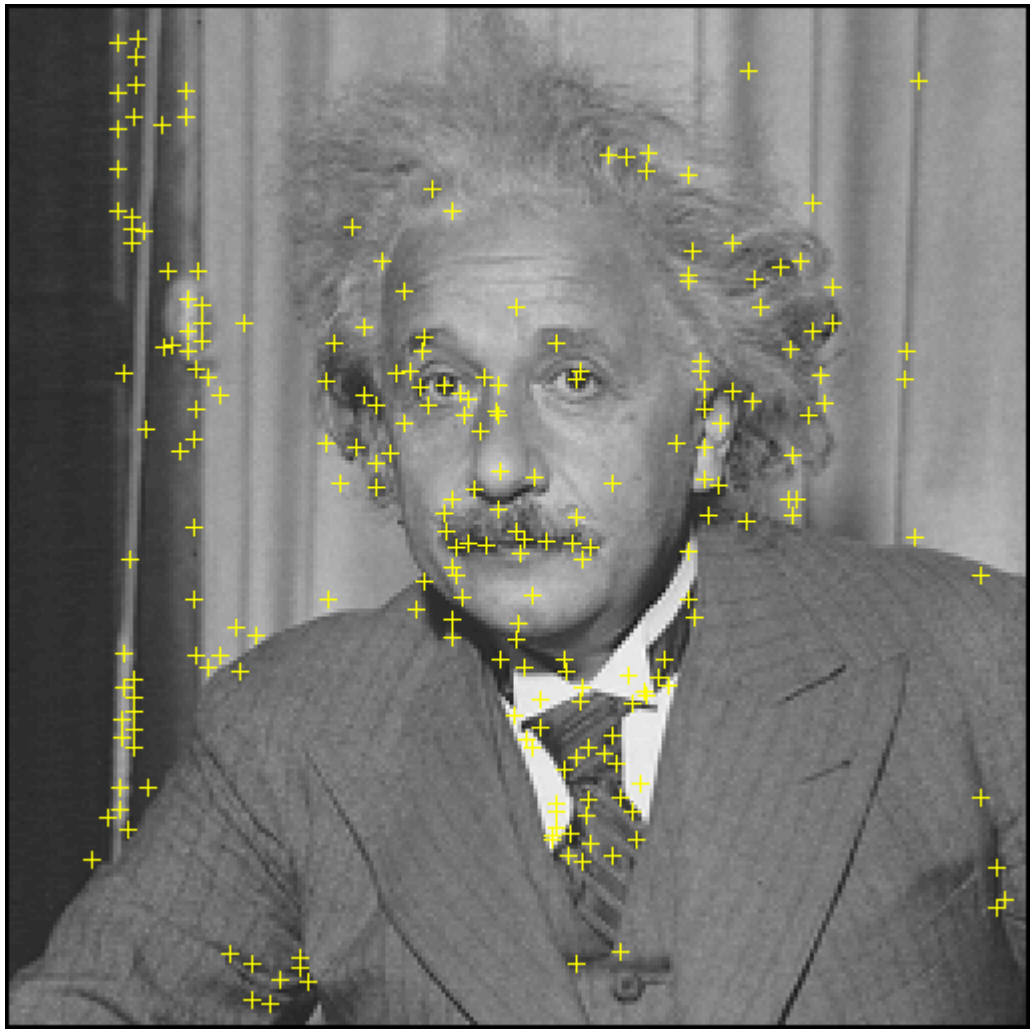


X est selectionné si valeur est inferieure ou superieure a ses 26 voisins

Maxima dans DoG

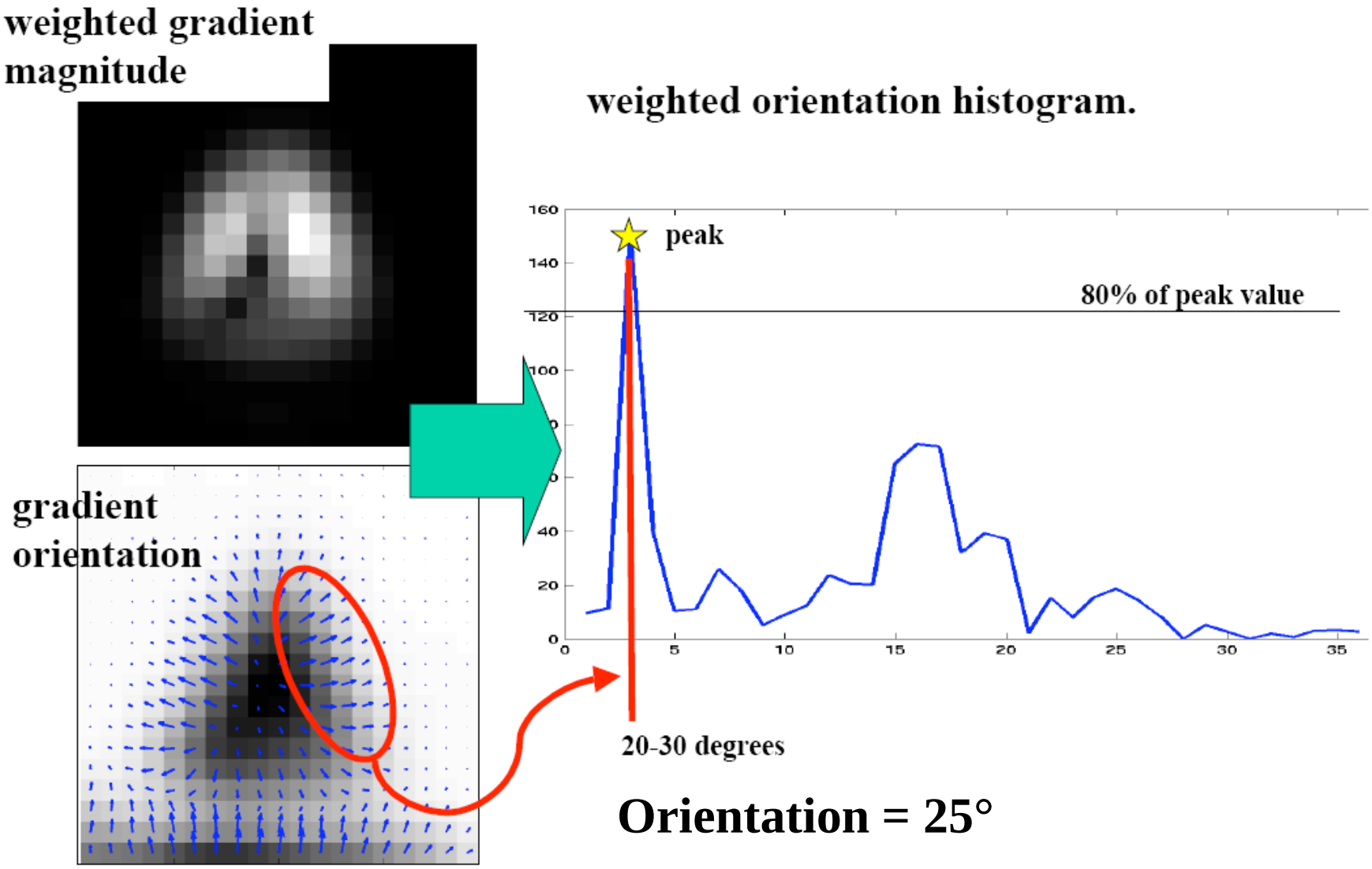


Filtrage des extrema

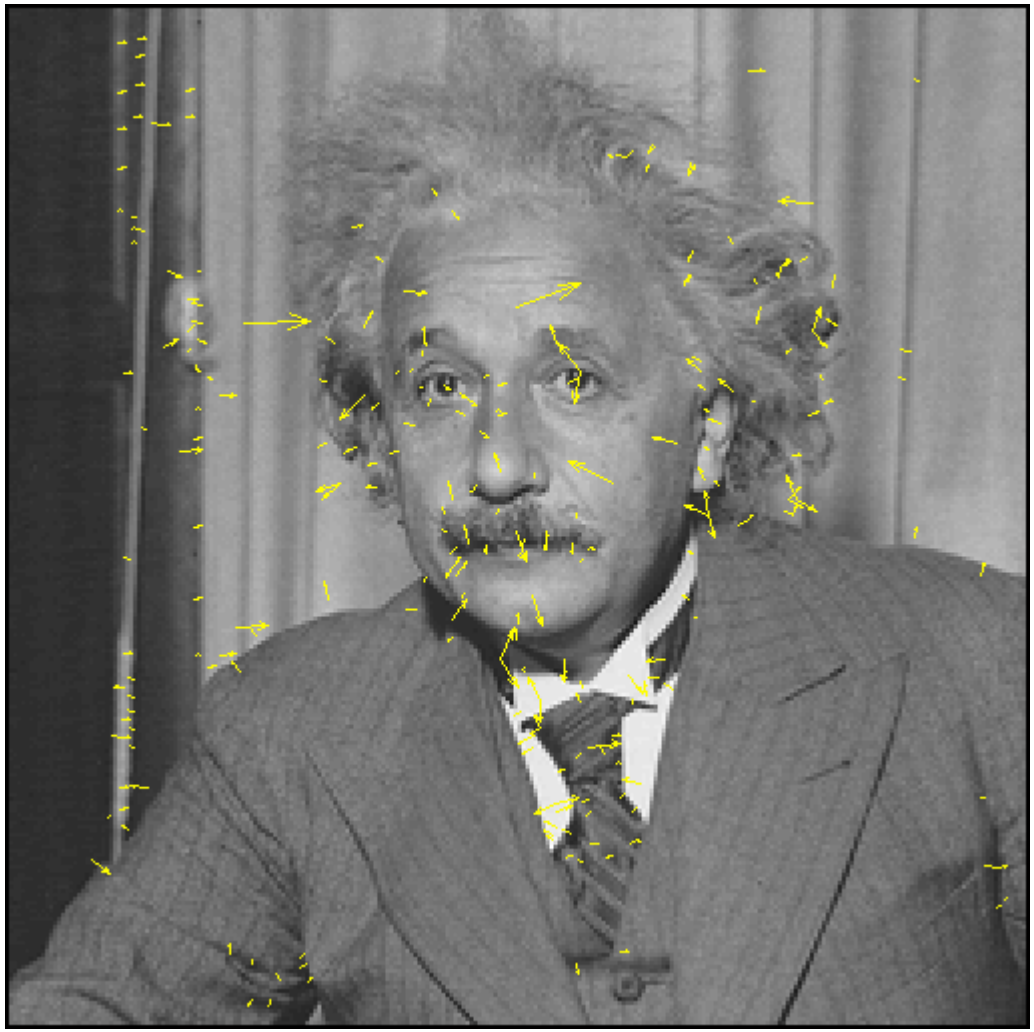


Suppression points de faible contraste et des points de contours

Detection de l'orientation

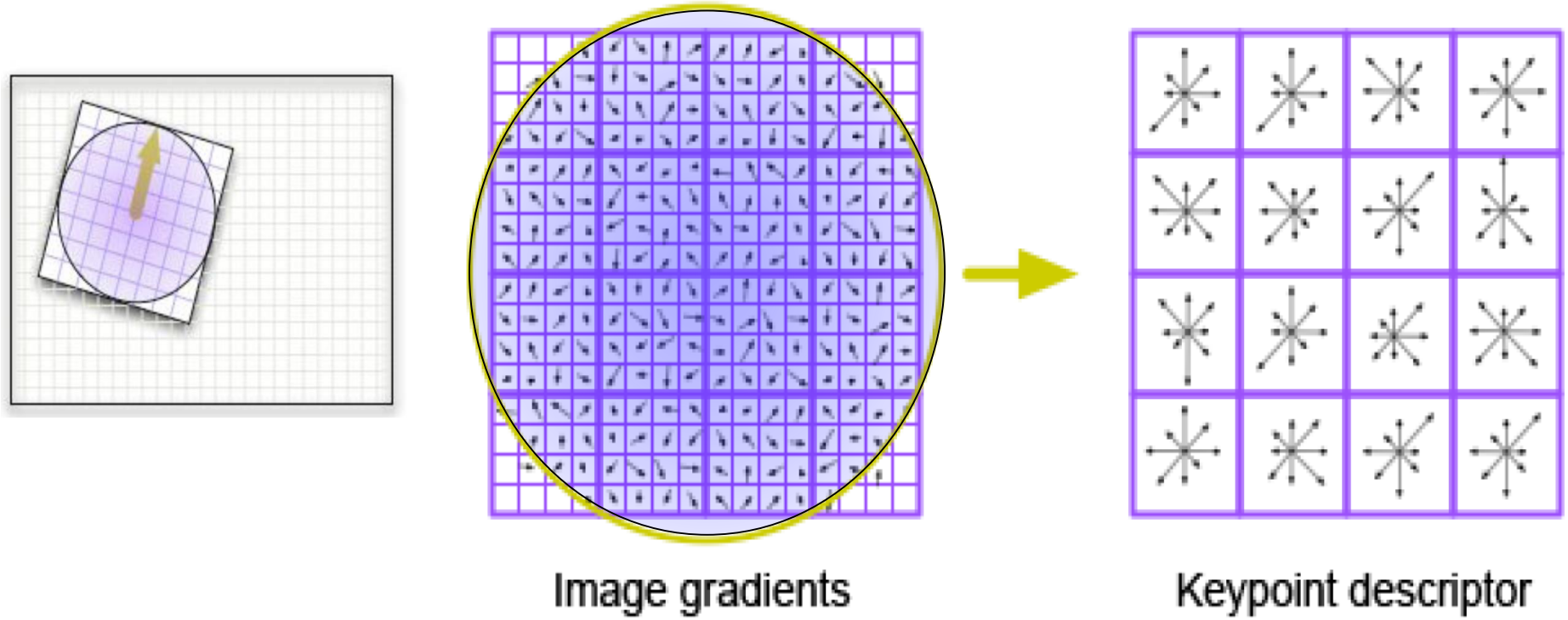


SIFT descripteur

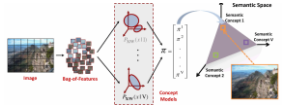


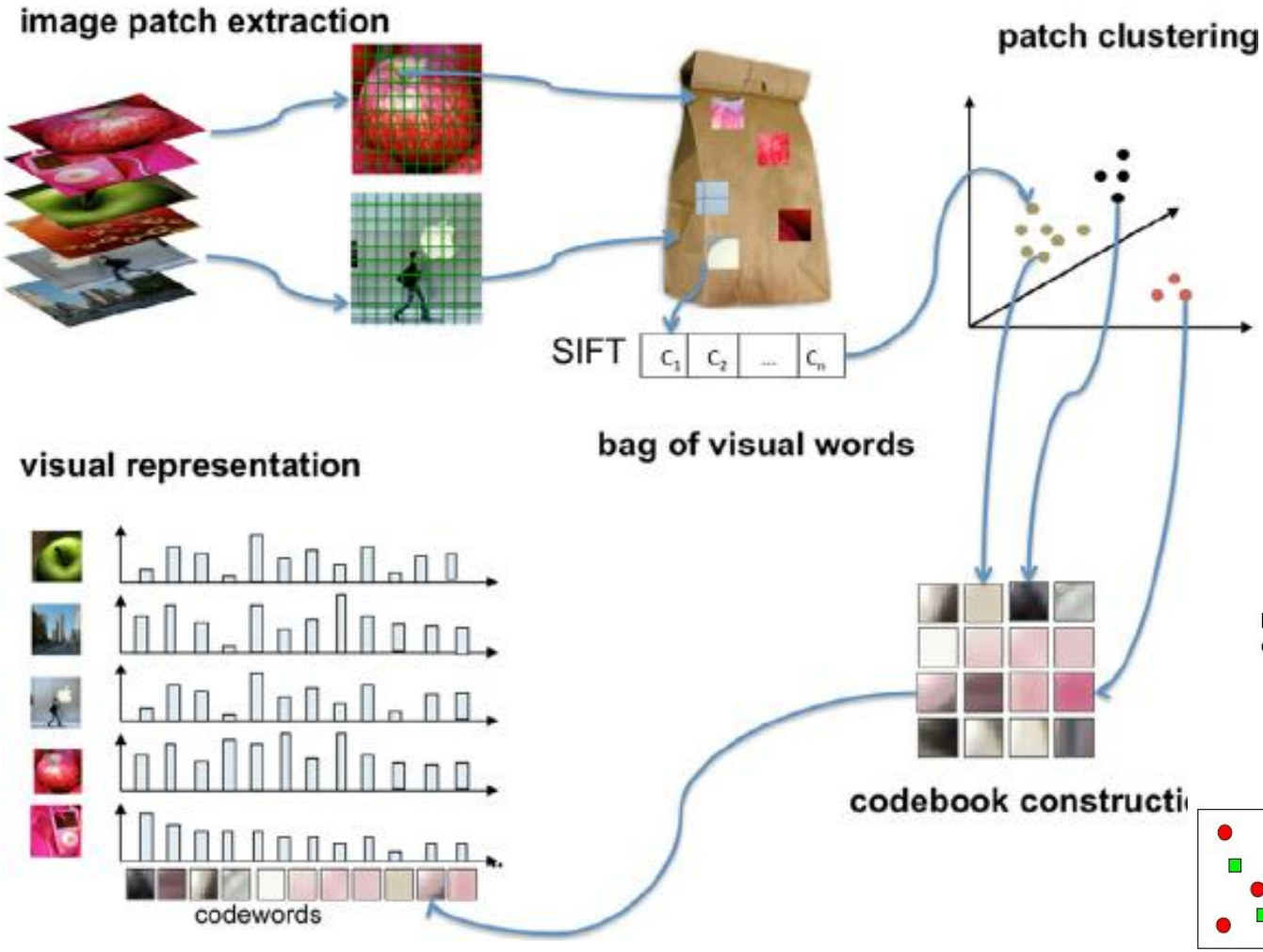
SIFT Descripteur

- 16 Histogrammes avec 8 directions (calculés dans 16 fenetres)
- Gradient principal calculé par application d'une grille 4x4 dans chaque fenetre
- Pondération Gaussienne autour du centre
- Vecteur de caractèristiques de 16 x 8 = 128 dimensions



Bag of Visual Features



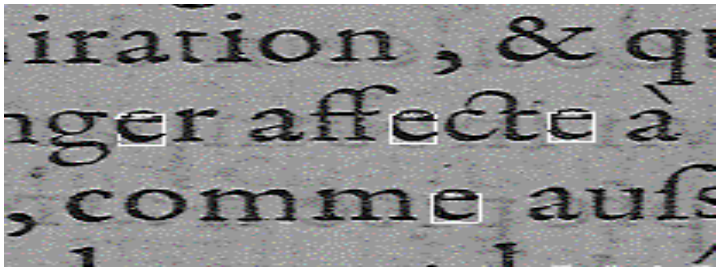


OCR : Un exemple de mise en application…

(ici, très, très, simplifié …)

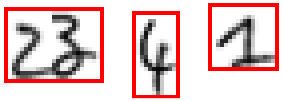
Des images à la RF : un exemple complet (simplifié)

1. Numérisation (scanner)



2. Prétraitement :
   - Supprimer les bruits, filtrer, ...

3. Segmenter pour isoler chaque caractère



3. Extraction de caractéristiques : créer un vecteur de caractéristiques pour chaque caractère
   - V(2,3 ; 2 ; 100; 50 ; … ; 45)
   - V(3 ; 10,2 ; 0 ; 20 ; … ; 5)

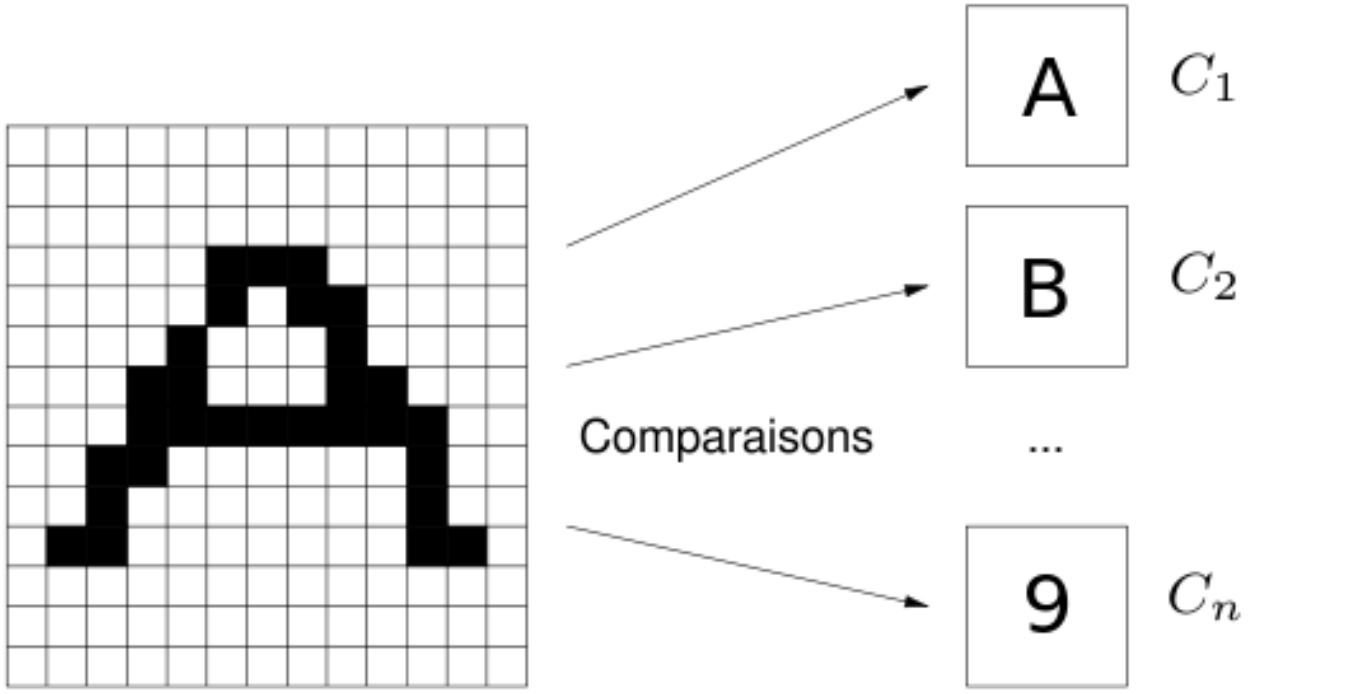
4. Classification : associer un symbole à un vecteur de mesures

5. Post-traitements : éventuellement utiliser le contexte pour corriger les erreurs éventuelles.

Extraction de caractéristiques 1/3

Comparaison directe

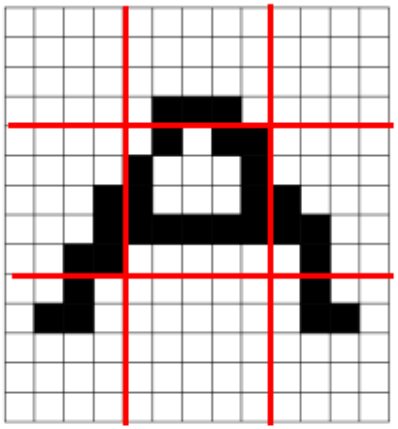
- distance(lettre; modele) = $\sum_{ij}$ ($P_{lettre}(i; j)$ - $P_{modele}(i; j)$)



Extraction de caractéristiques 2/3

Zoning

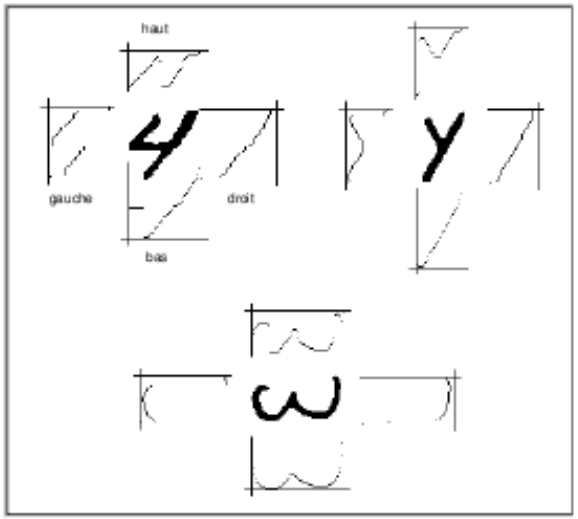
- Découpage en n blocs de l'image du caractère
- Pour chaque bloc on calcule une caractéristique (ici le nombre de pixels noirs)
- I vecteur de caractéristiques : V = (d1; d2; …..; dn)

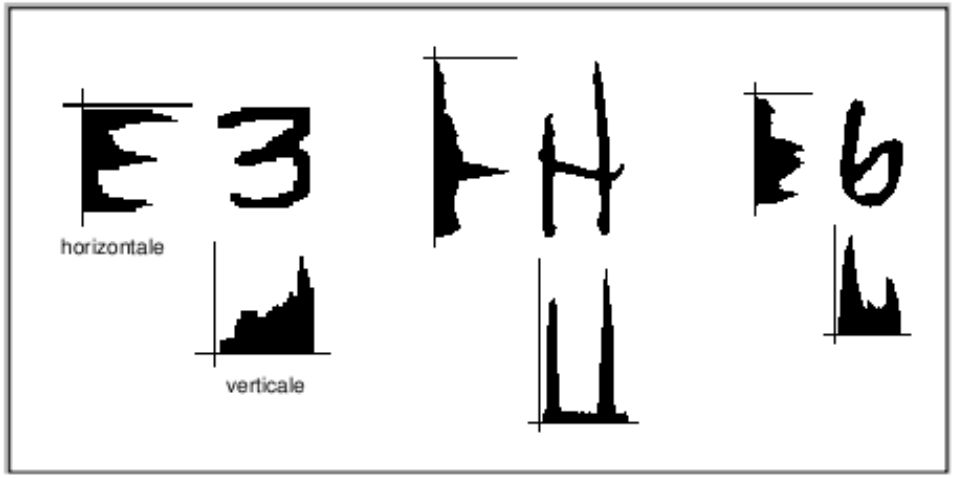


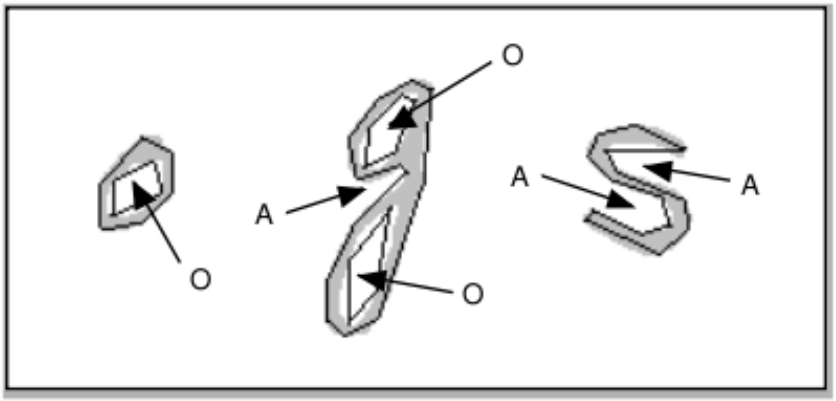
V=(0,3,0,4,12,4,3,0,3)

Extraction de caractéristiques 3/3

Calcul de profils et de courbures







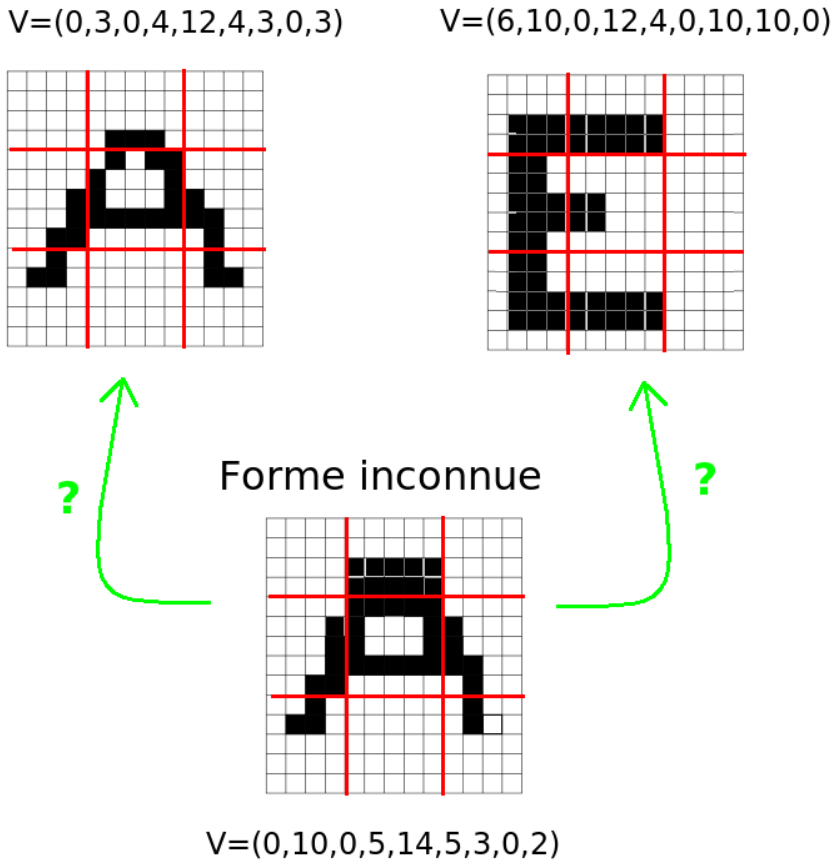
Apprentissage & classification

1. Phase d'apprentissage :
   - Créer une base d'apprentissage manuellement → étiqueter des données (labéliser des exemples)
   - Utiliser la base d'apprentissage pour apprendre un modèle
   - Un modèle simplifié → Chaque classe est défini par 1 seul exemple pris dans la base ou le vecteur moyen, ou médian...

2. Phase de classification : trouver la classe de l'objet inconnu représenté par un vecteur de caractéristiques X
   - 1-ppv → classification basée sur une mesure de distance (euclidienne,...)
   - La classe attribuée est celle du vecteur de référence le plus proche

Exemple de classification d'une forme inconnue

- La forme est donc identifiée comme étant un "A" car
- min(D(A; ?);D(B; ?)) = D(A; ?)



$$D(A,?) = \sqrt{(0-0)^2 + (3-10)^2 + (0-0)^2 + (4-5)^2 + \ldots + (3-2)^2}$$

$$D(A,?) = 7,48 \text{ et } D(B,?) = 19,05$$

BILAN ? ? ?

- Votre avis m’intéresse…

- Prochaines séances : Intervenant extérieur

Forte présence et motivation appréciée

MERCI