Robotique Mobile
13 – SLAM 01 – EKF SLAM

David Filliat

Alexandre Chapoutot

Goran Frehse

prenom.nom@ensta-paris.fr

Cartographie

recherche de la carte *la plus probable*, connaissant les données perçues par le robot, dans l'espace des cartes de la représentation choisie

→ Espace beaucoup plus grand que l'espace utilisé pour la localisation

Localisation associée à la cartographie

→ recherche dans un espace ouvert

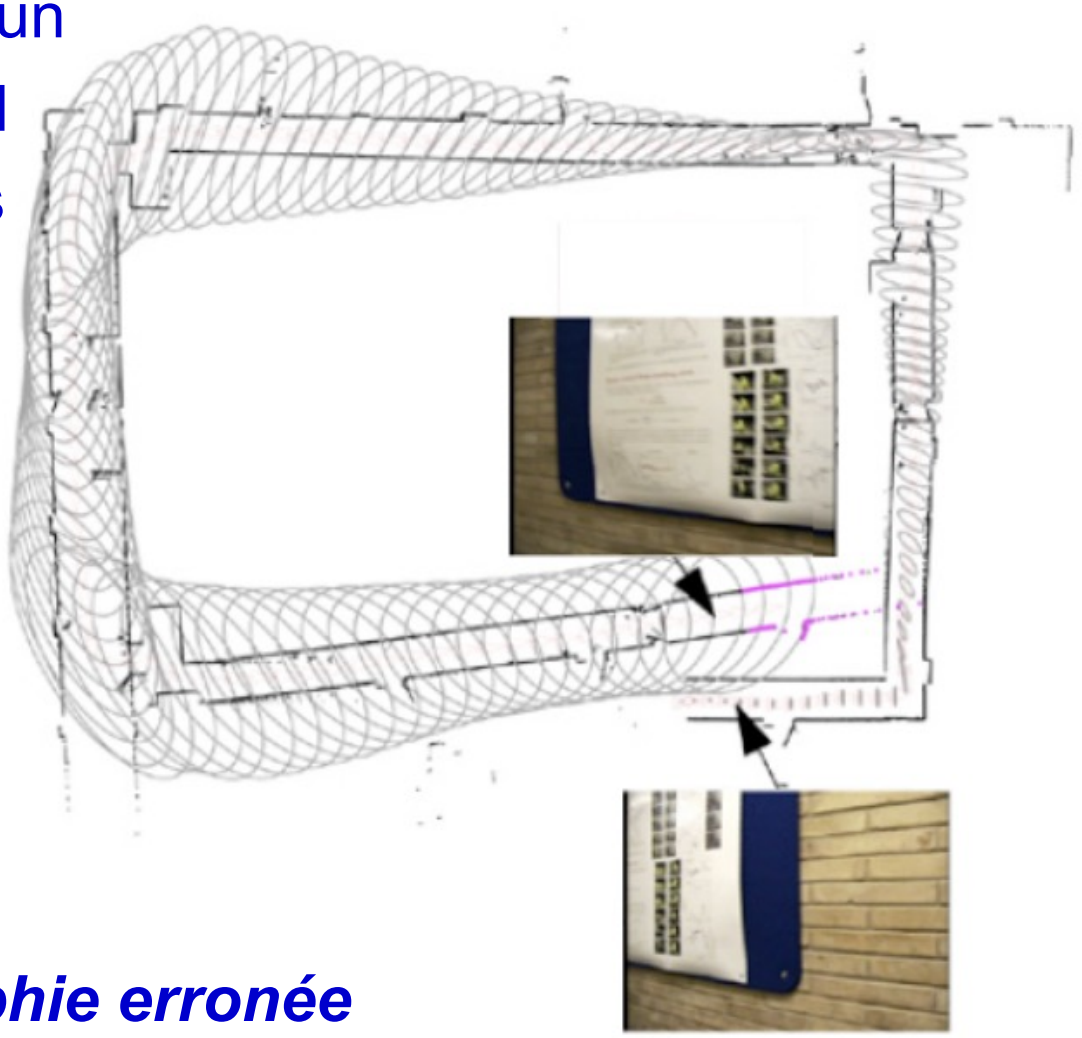
Carte incomplète

→ Utilisation principalement de suivi de position en découvrant l'environnement

→ limitations, notamment pour les grands cycles

Environnement contenant un cycle beaucoup plus grand que la portée des capteurs

Dérive des méthodes de localisation (suivi de position)



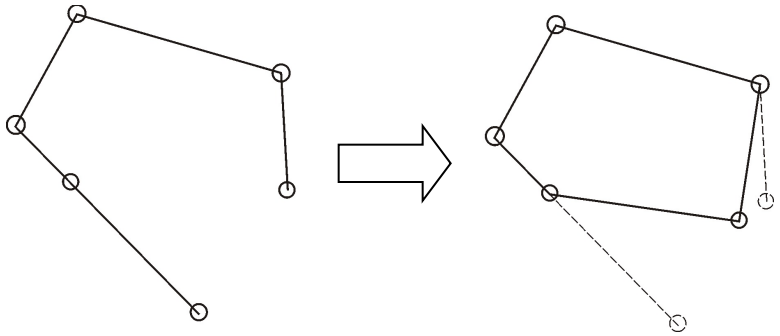
→ ***cartographie erronée***

P. Newman, Oxford Mobile Robotics Group

Cartographie incrémentale

– Pas de reprise des informations passées si elle se révèlent erronées

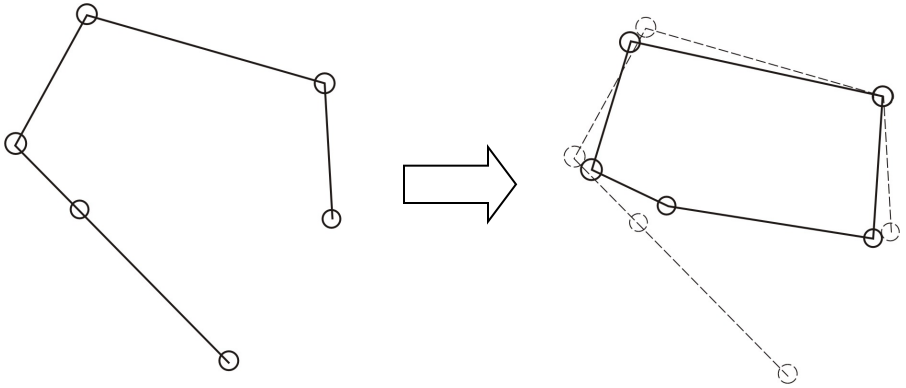
→ modifications locales de la carte



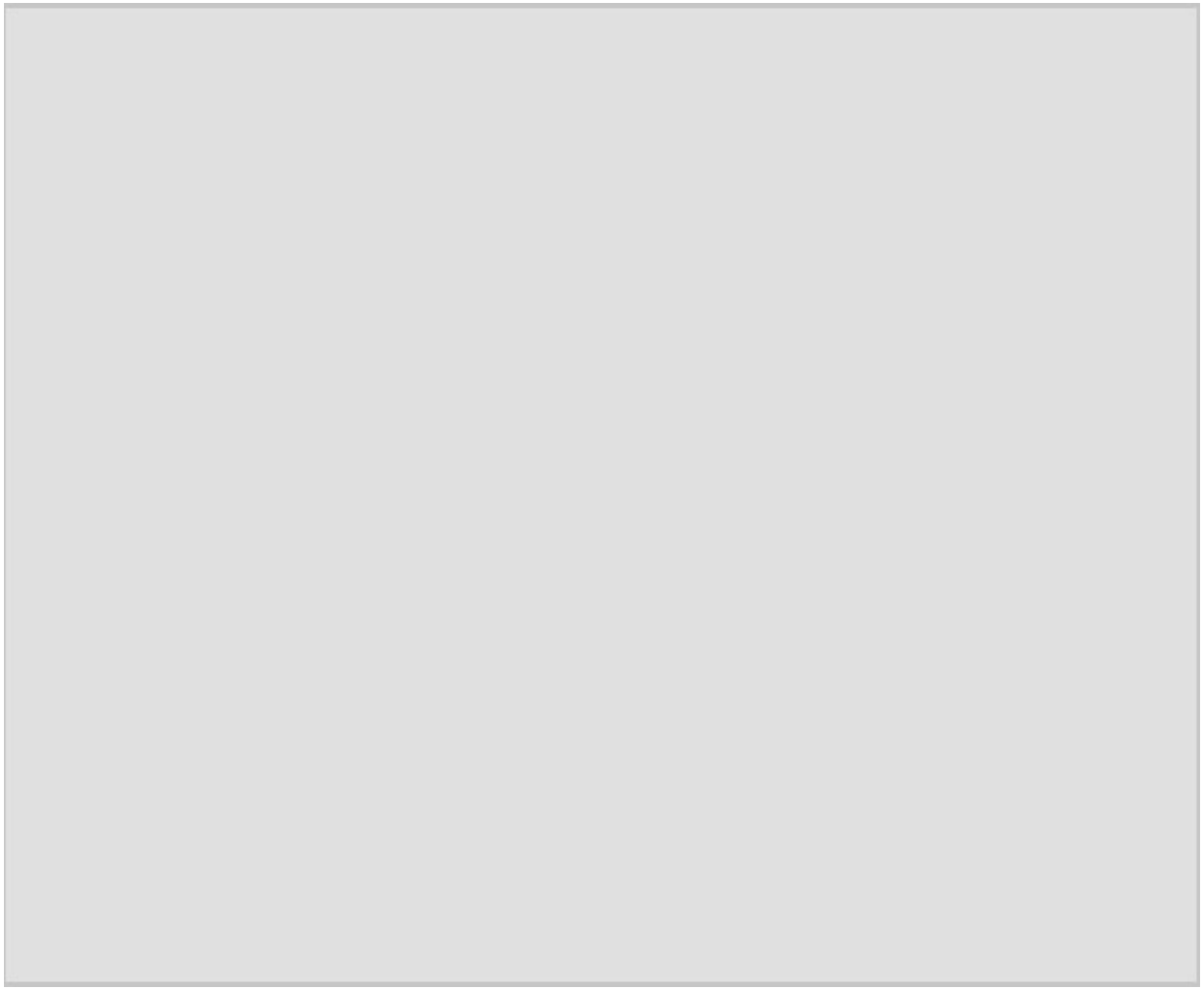
Cartographie avec retour en arrière

– Lorsque la localisation se révèle fausse, il existe une méthode pour reprendre les modifications déjà faites

→ possibilité de modifications globales



Raw Data



Robotics and **S**tate **E**stimation Lab (Dieter Fox)

http://rse-lab.cs.washington.edu/

Cartographie
avec retour arrière

Robotics and **S**tate **E**stimation Lab (Dieter Fox)

http://rse-lab.cs.washington.edu/

Cartographie incrémentale

Cartographie topologique

– Algorithme simple, dépend beaucoup de la qualité de la localisation (essentiellement par les perceptions)

Algorithm 4 Algorithme de cartographie topologique pour un déplacement u et des perceptions Y

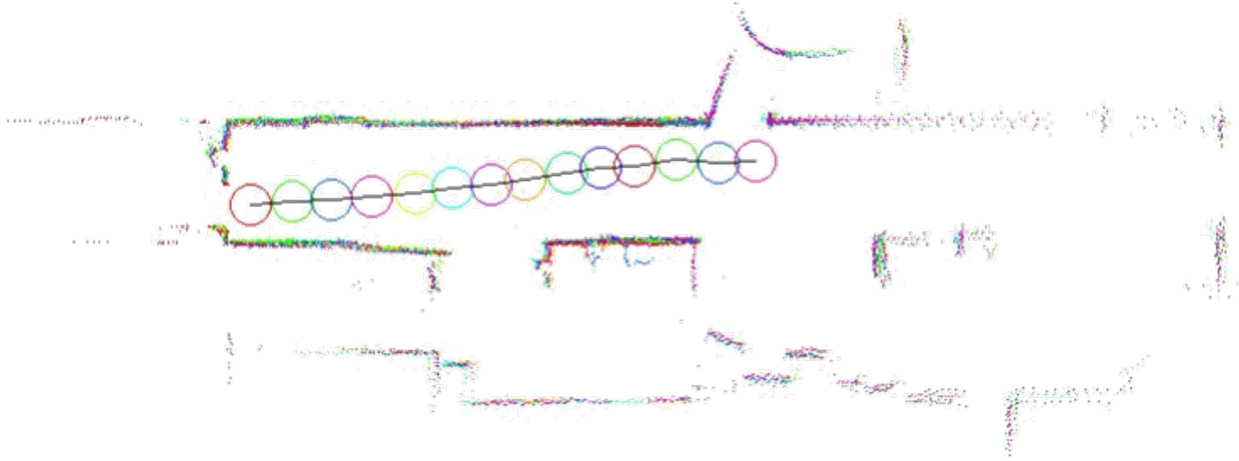
1: **if** Il existe $noeud_i$ compatible avec u, Y et $Position_{t-1}$ **then**
2: $Position_t = noeud_i$
3: **else**
4: $Position_t$ = nouveau noeud
5: **end if**
6: Mettre à jour données de $Position_t$ avec Y
7: Mettre à jour la connexion $Position_{t-1}$ - $Position_t$ avec u

Corrélation de scans

- Nombreux algorithmes, par exemple ICP
- Utilisable en environnement inconnu
- Hyp : large recouvrement entre scans successifs

Algorithme

- Corrélation avec le scan précédent -> localisation
- Corrélation avec le scan de la carte le plus proche
  -> recalage lors de fermetures de boucles
- Ajout du scan à la carte si on est assez loin de tous les autres



Limitation pour les grands cycles

- Localisation corrigée, mais pas la carte lors de fermetures de boucles
- Gestion des fermetures de boucles possible (CF plus loin)

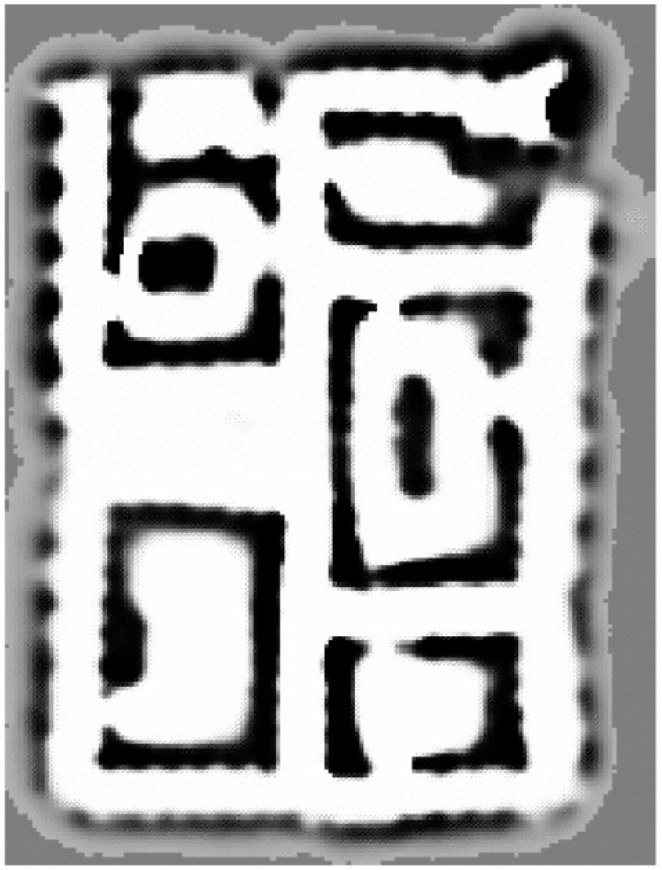
Probabilité de présence d'obstacles

- Représentée sur grille régulière

Position *supposée connue*

ou Position estimée par suivi de position

- Corrélation de scans
- Corrélation de grille locale avec la grille globale
- Hypothèses sur l'environnement (murs //)



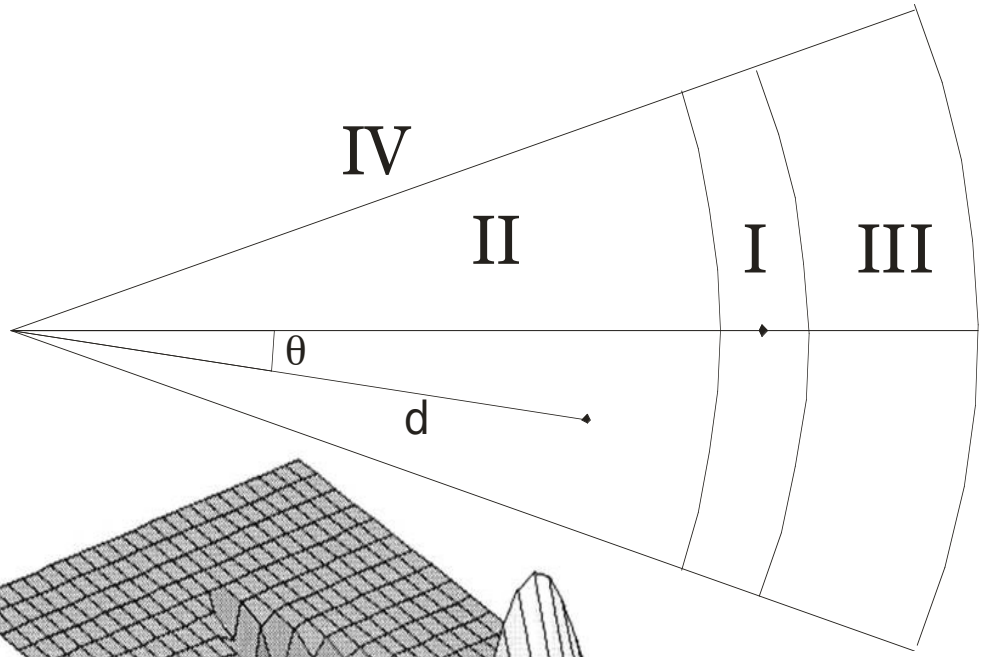
On estime : $P(occ_i | s_1, \ldots, s_T)$

en connaissant : $P(occ_i | s)$ (modèle inverse)

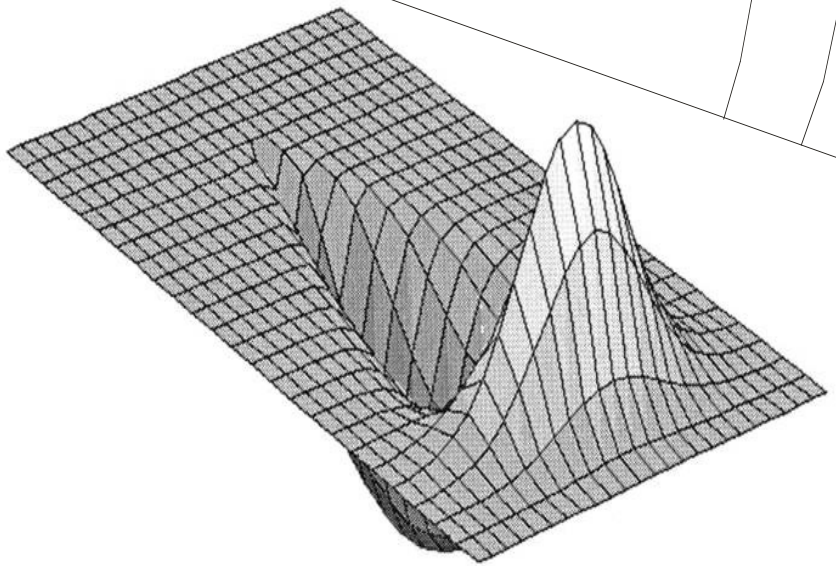
Position connue !

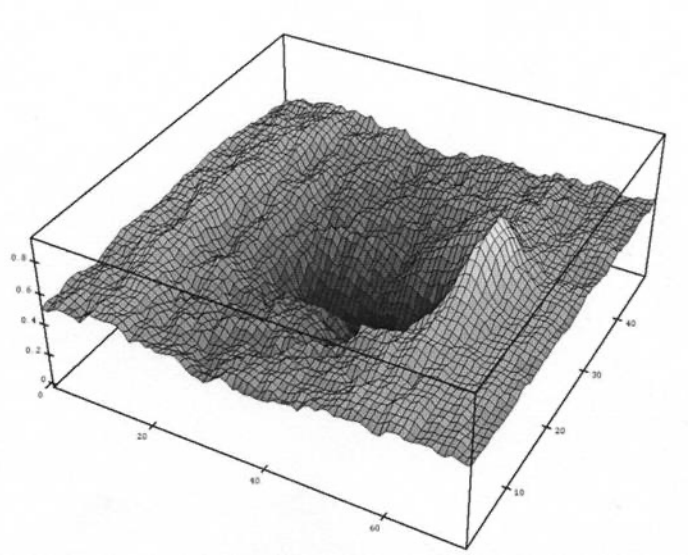
Rappel

– P(valeur réelle|mesure) :

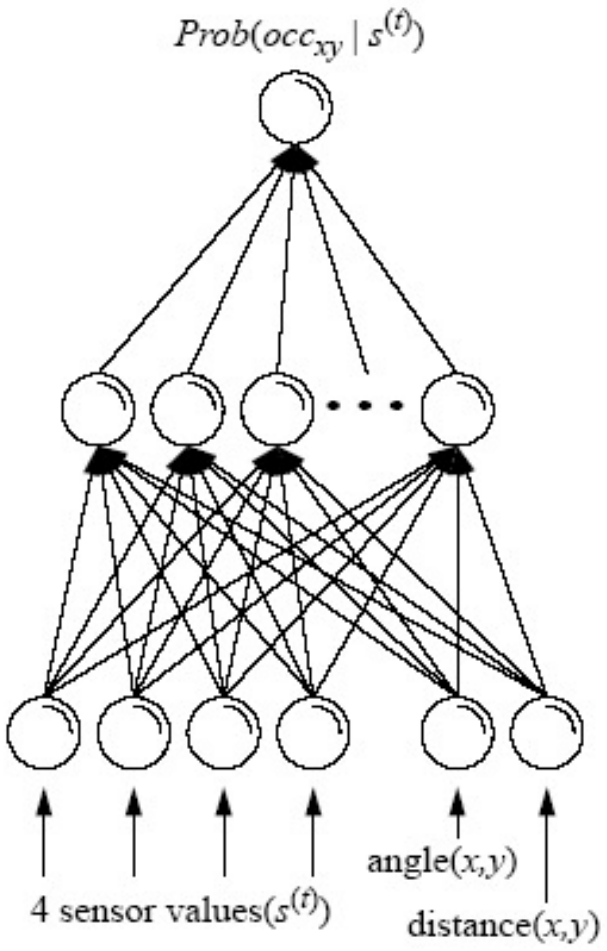


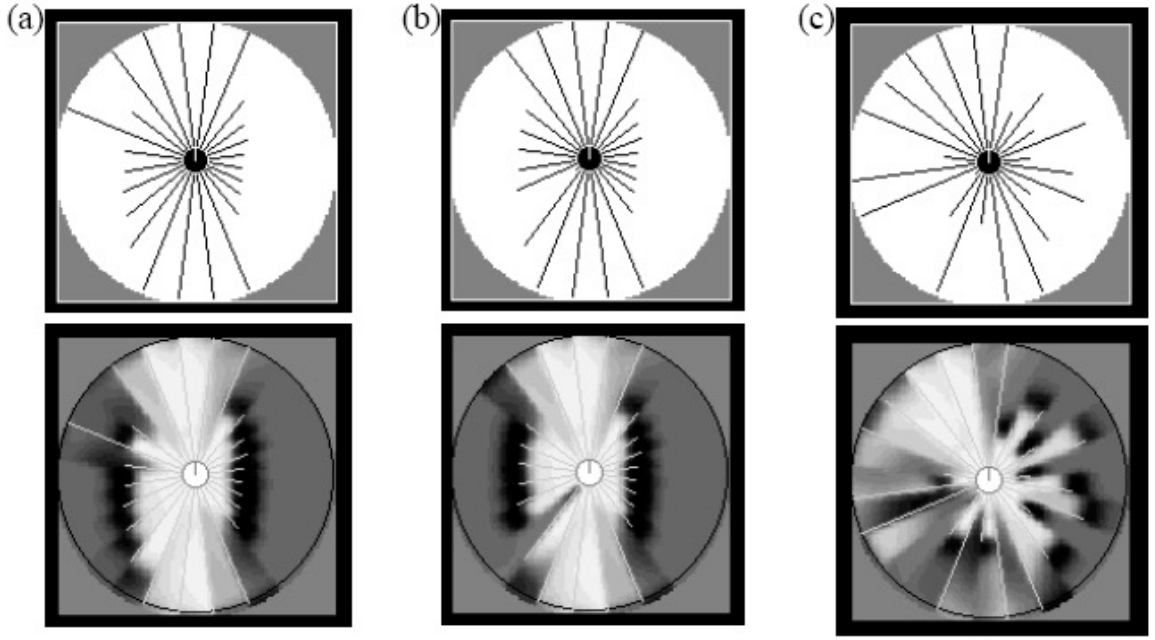
I : Gauss(θ)*Gauss(d-d_m)
II : exp(d)*exp(-θ)
III : exp(d)*exp(-θ)
IV : 0





Variante : Réseau de neurone utilisant plusieurs mesures (Thrun 98) :





Mise à jour de la probabilité d'une cellule dans le champ du capteur :

- Utilisation d'une valeur intermédiaire:

$$l_i^T = log\left(\frac{P(occ_i^T)}{1-P(occ_i^T)}\right)$$

Le log permet de stocker des probabilité faibles avec moins d'approximations

- Mise à jour :

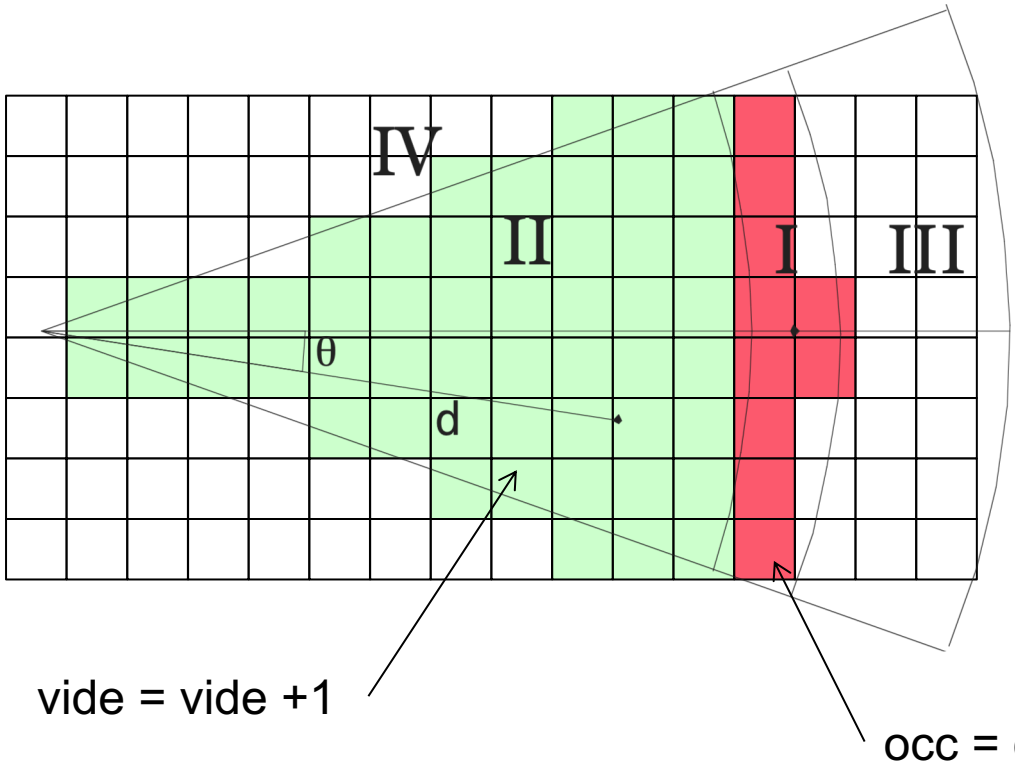
$$l_i^T = log\left(\frac{P(occ_i|s_T)}{1-P(occ_i|s_T)}\right) + l_i^{T-1}$$

- éventuellement :

$$p(occ_i^T) = 1 - \frac{1}{e^{l_i^T}}$$

Variante : simple décompte

$$P_{occ}(x,y) = \frac{occ(x,y)}{vide(x,y) + occ(x,y)}$$

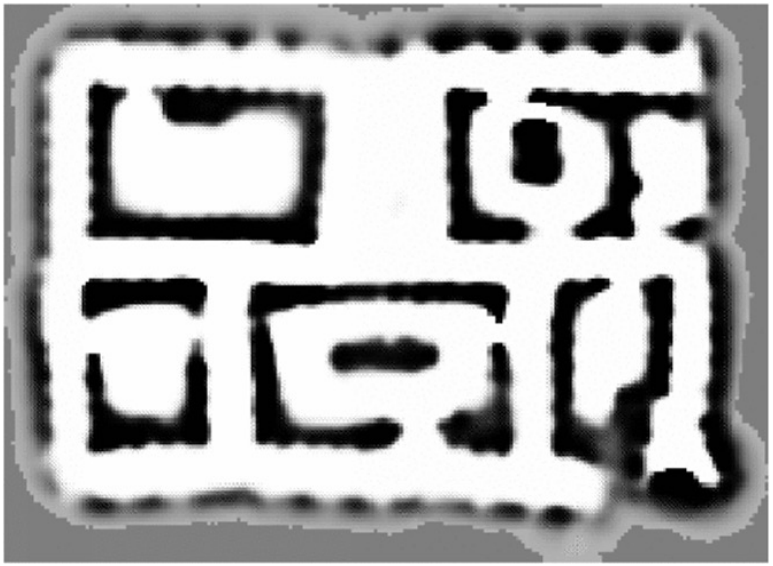


Exemples :

Utilisation de sonars

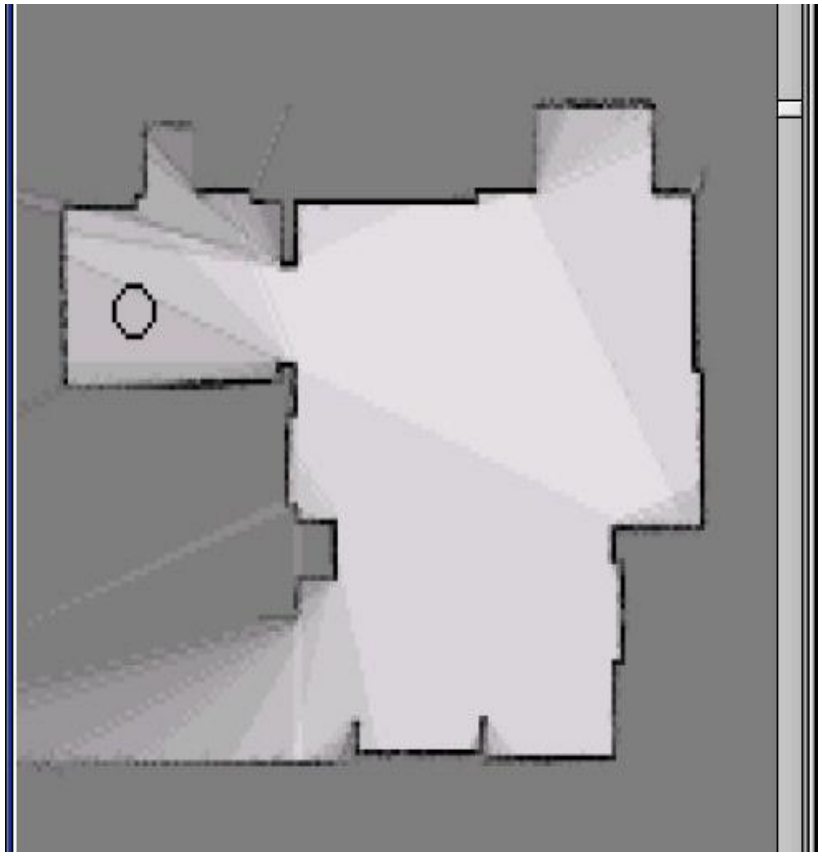
+ hypothèse murs orthogonaux

(Thrun 98)



Utilisation d'un télémètre laser

+ localisation par corrélation de scans



Autre exemple : HECTOR SLAM

Utilisation d'un télémètre laser

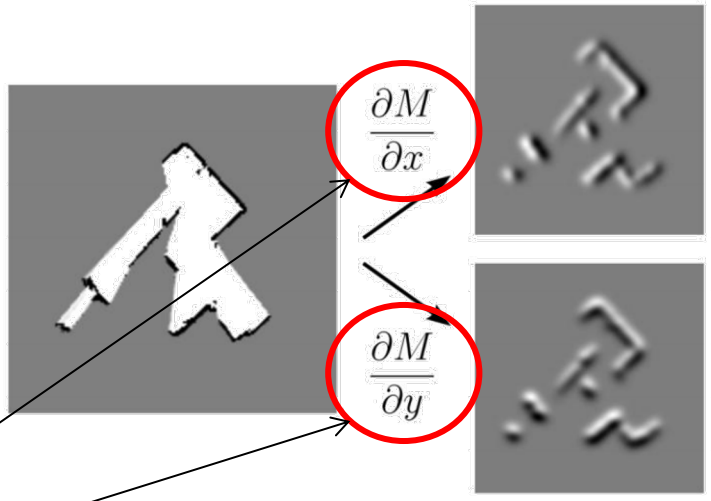
Localisation par rapport à la carte par descente de gradient

– Trouver position tq tt les pts du laser soient dans des cases occupées:

$$\boldsymbol{\xi}^* = \underset{\boldsymbol{\xi}}{\operatorname{argmin}} \sum_{i=1}^{n} \left[1 - M(\mathbf{S}_i(\boldsymbol{\xi}))\right]^2$$

– Dvp de Taylor ordre 1 :

$$\sum_{i=1}^{n} \left[1 - M(\mathbf{S}_i(\boldsymbol{\xi} + \Delta\boldsymbol{\xi}))\right]^2 \rightarrow 0$$



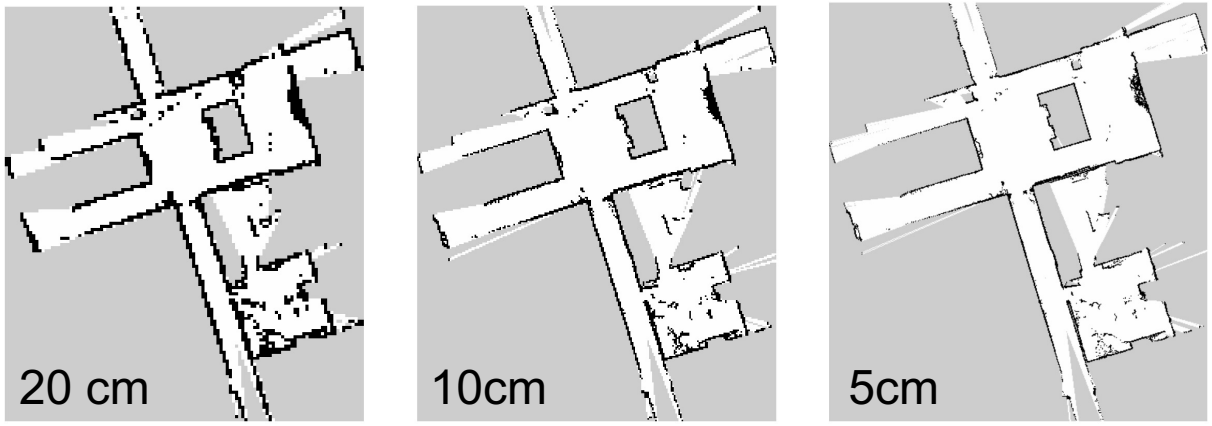
– Méthode de Gauss-Newton

$$\Delta\boldsymbol{\xi} = \mathbf{H}^{-1} \sum_{i=1}^{n} \left[\nabla M(\mathbf{S}_i(\boldsymbol{\xi})) \frac{\partial \mathbf{S}_i(\boldsymbol{\xi})}{\partial \boldsymbol{\xi}}\right]^T \left[1 - M(\mathbf{S}_i(\boldsymbol{\xi}))\right]$$

Autre exemple : HECTOR SLAM

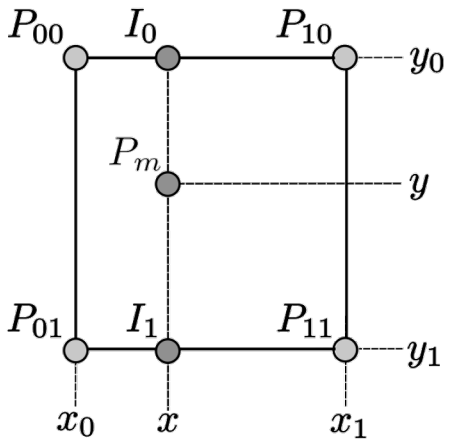
Quelques trucs supplémentaires:

Approche multi-échelle



Estimation continue via interpolation.

$$M(P_m) \approx \frac{y - y_0}{y_1 - y_0}\left(\frac{x - x_0}{x_1 - x_0} M(P_{11}) + \frac{x_1 - x}{x_1 - x_0} M(P_{01})\right) + \frac{y_1 - y}{y_1 - y_0}\left(\frac{x - x_0}{x_1 - x_0} M(P_{10}) + \frac{x_1 - x}{x_1 - x_0} M(P_{00})\right)$$



Autre exemple : HECTOR SLAM

S. Kohlbrecher and J. Meyer and O. von Stryk and U.Klingauf : A Flexible and Scalable SLAM System with Full 3D Motion Estimation.

ROS package Hector_SLAM

Limitations

- Perceptual aliasing -> topologie/structure incohérente
- Nécessite une localisation métrique précise
- Limité pour les environnements contenant de grands cycles

  $\Rightarrow$ fermetures non détectée

  $\Rightarrow$ erreurs lors de la fermeture des cycles

- Impossibilité de retours en arrière sur des mises à jour déjà faites

mais Modifications possibles de la carte selon nouvelles données

Cartographie avec retour en arrière

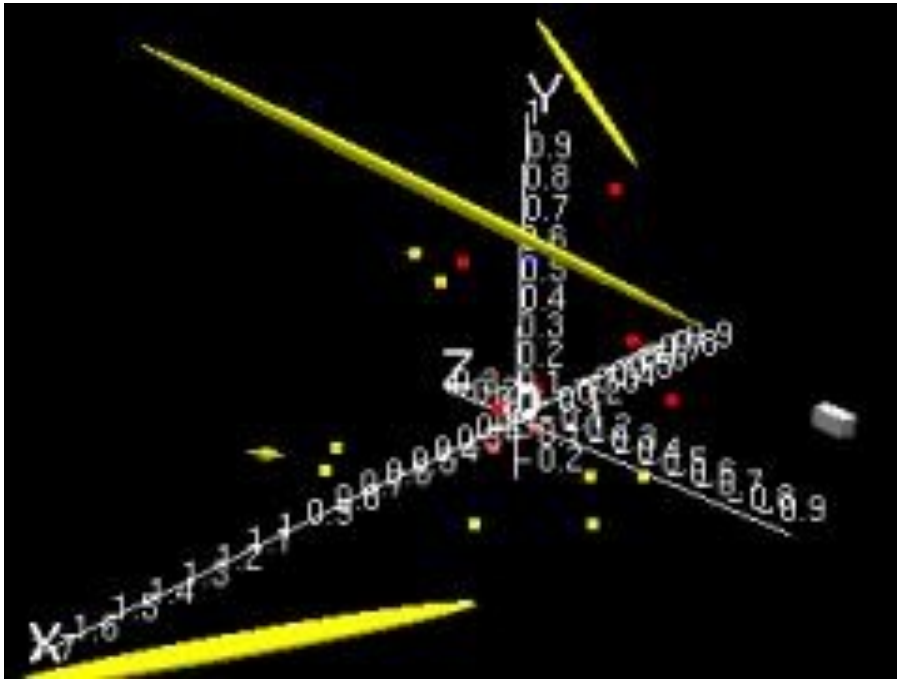
Méthodes de filtrage

Objectif :

- propager de nouvelles informations sur la position courante aux positions précédentes

→ Mémorisation des relations positions du robot / éléments de la carte pour pouvoir les ré-utiliser

→ Utilisation de carte d'amers ou d'une carte de scans



Cadre général

Filtrage Bayésien sur la position du robot et des éléments de la carte:

$$Bel(x_t, c_t) = \eta p(y_t|x_t, c_t) \int\int p(x_t, c_t|x_{t-1}, c_{t-1}, u_{t-1}) Bel(x_{t-1}, c_{t-1}) dx_{t-1} dc_{t-1}$$

Carte statique :

$$Bel(x_t, c_t) = \eta p(y_t|x_t, c_t) \int\int p(x_t|x_{t-1}, u_{t-1}) Bel(x_{t-1}, c_{t-1}) dx_{t-1} dc_{t-1}$$

Estimation complète de Bel(x,c)

→ approximations

- Filtrage de Kalman étendu
- Filtrage Particulaire
- …

EKF Simultaneous Localization and Mapping

- Filtre de Kalman étendu
- État = position du robot et des éléments de l'environnement (amers, scans..)

$$Bel(x_t, c_t) = \begin{bmatrix} x \\ y \\ \theta \\ x_{a1} \\ y_{a1} \\ \vdots \\ x_{aN} \\ y_{aN} \end{bmatrix}$$

Robot

Carte

Rappel : Filtre de Kalman Etendu

$$\begin{aligned} X_{t+1} &= f(X_t, u_t) + \varepsilon_{evo} \\ Y_t &= h(X_t) + \varepsilon_{obs} \end{aligned}$$

$$A_{ij} = \frac{\partial f_i}{\partial x_j}$$

$$B_{ij} = \frac{\partial f_i}{\partial u_j}$$

$$H_{ij} = \frac{\partial h_i}{\partial x_j}$$

$$\begin{aligned} X_t^* &= f(X_{t-1}, u_t) \\ P_t^* &= A.\hat{P}_{t-1}.A^T + B.Q.B^T \\ Y_t^* &= h(X_t^*) \\ K &= P_t^* H^T.(H.P_t^*.H^T + P_Y)^{-1} \\ \hat{X}_t &= X_t^* + K(Y_t - Y_t^*) \\ \hat{P}_t &= P_t^* - KHP_t^* \end{aligned}$$

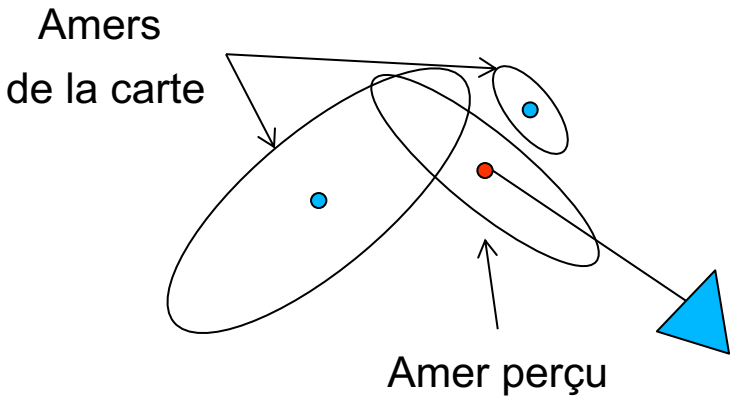
- Les jacobiennes se font aussi par rapport à la position des amers
- La mise à jour modifie à la fois la position du robot et des amers

Gestion des amers

- Pour tout amer perçu
  - S'il est dans la carte : utilisation du filtre pour estimer sa position et celle du robot en fonction des covariances

    → ~ Kalman pour la localisation
  - Si il n'est pas dans la carte : ajout

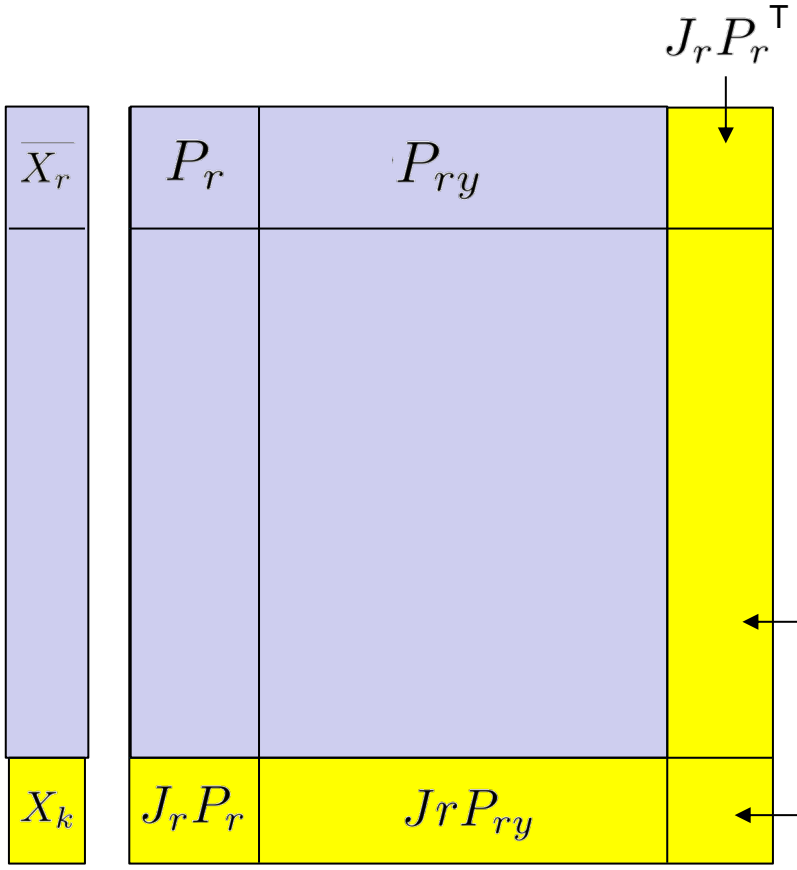
- Pour savoir s'il est dans la carte
  - amers uniques (caractérisation par les perceptions)
  - distance de Mahalanobis en cas de perceptual aliasing



$$d^2 = \frac{1}{2}(X-Y)^T(P_X+P_Y)^{-1}(X-Y)$$

Ajout des amers

- Agrandir X avec la position absolue
- Agrandir P avec la covariance

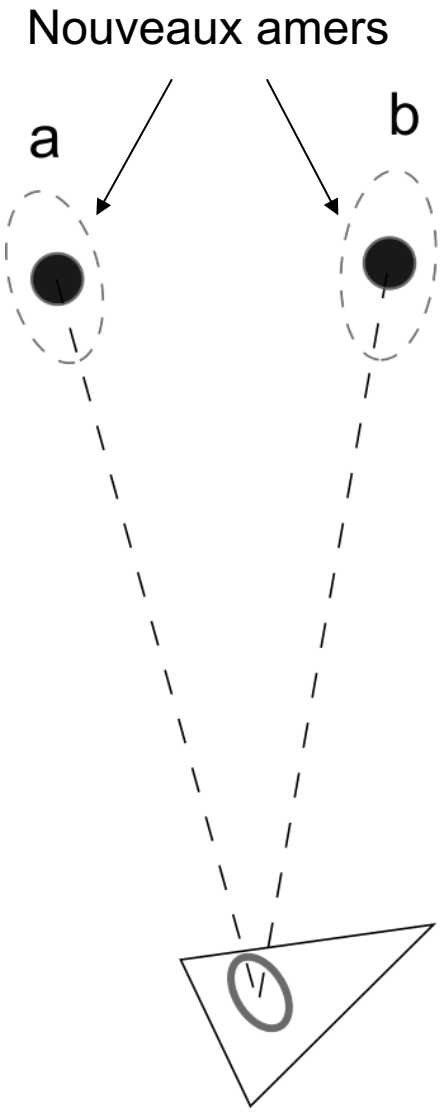


$$X_k = \begin{bmatrix} x_r + d_k \cos(\theta_r + \phi_r) \\ y_r + d_k \sin(\theta_r + \phi_r) \end{bmatrix}$$

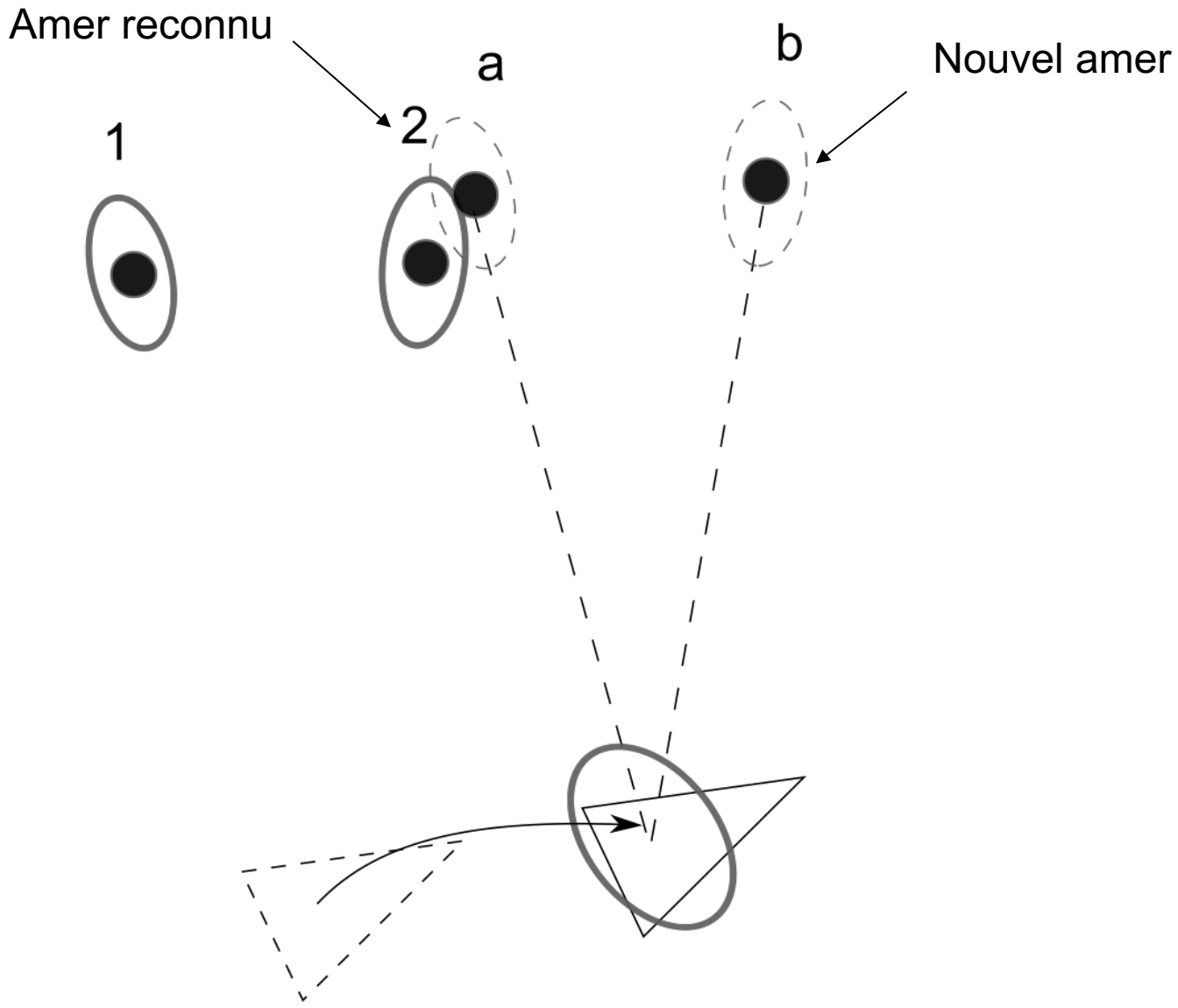
$$J_r = \frac{\partial X_k}{\partial X_r} = \begin{bmatrix} 1 & 0 & -d\sin(\theta+\phi) \\ 0 & 1 & d\cos(\theta+\phi) \end{bmatrix}$$

$$J_y = \frac{\partial X_k}{\partial Y} = \begin{bmatrix} \cos(\theta_r + \phi_r) & -d\sin(\theta+\phi) \\ \sin(\theta_r + \phi_r) & d\cos(\theta+\phi) \end{bmatrix}$$

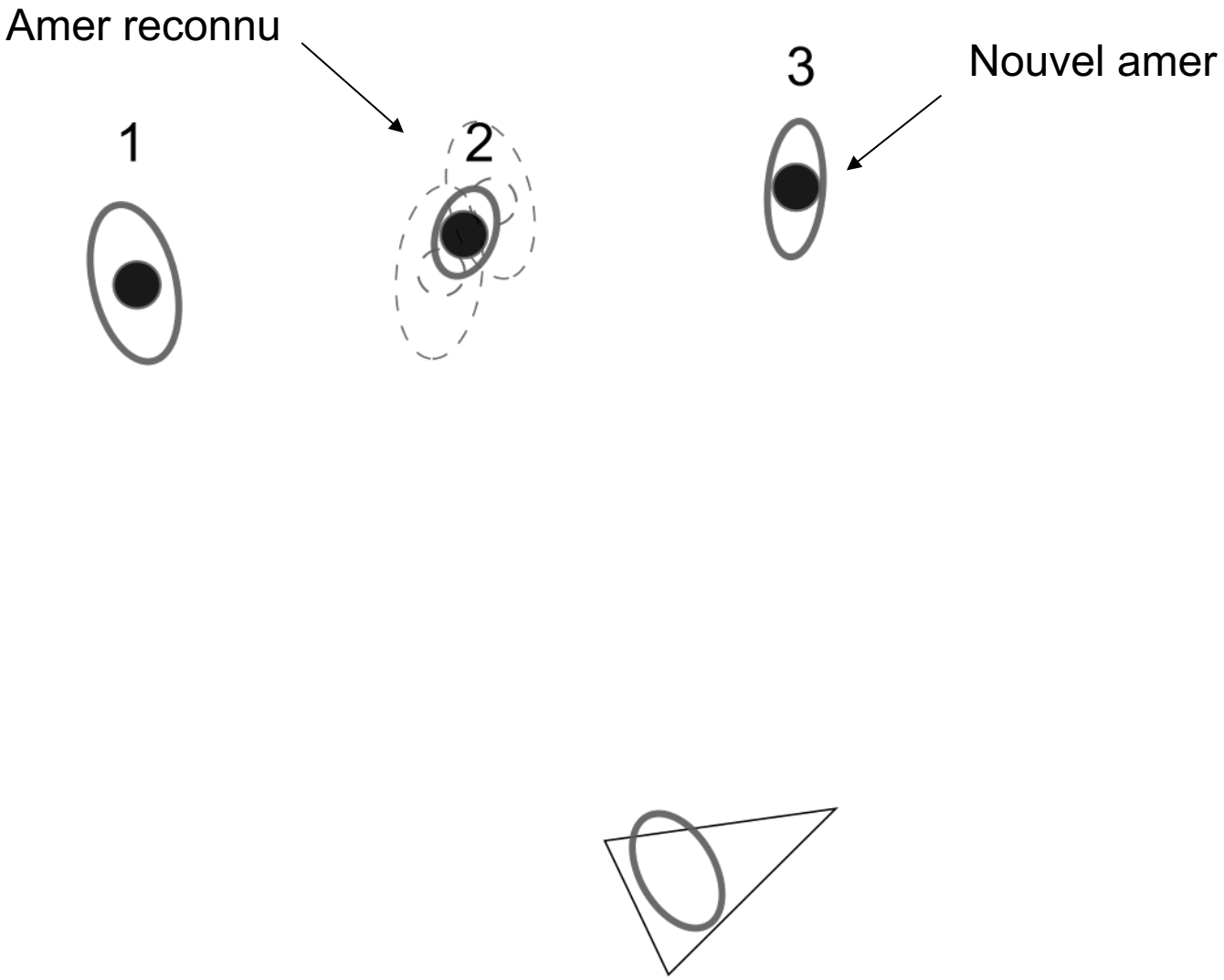
EKF SLAM – Illustration



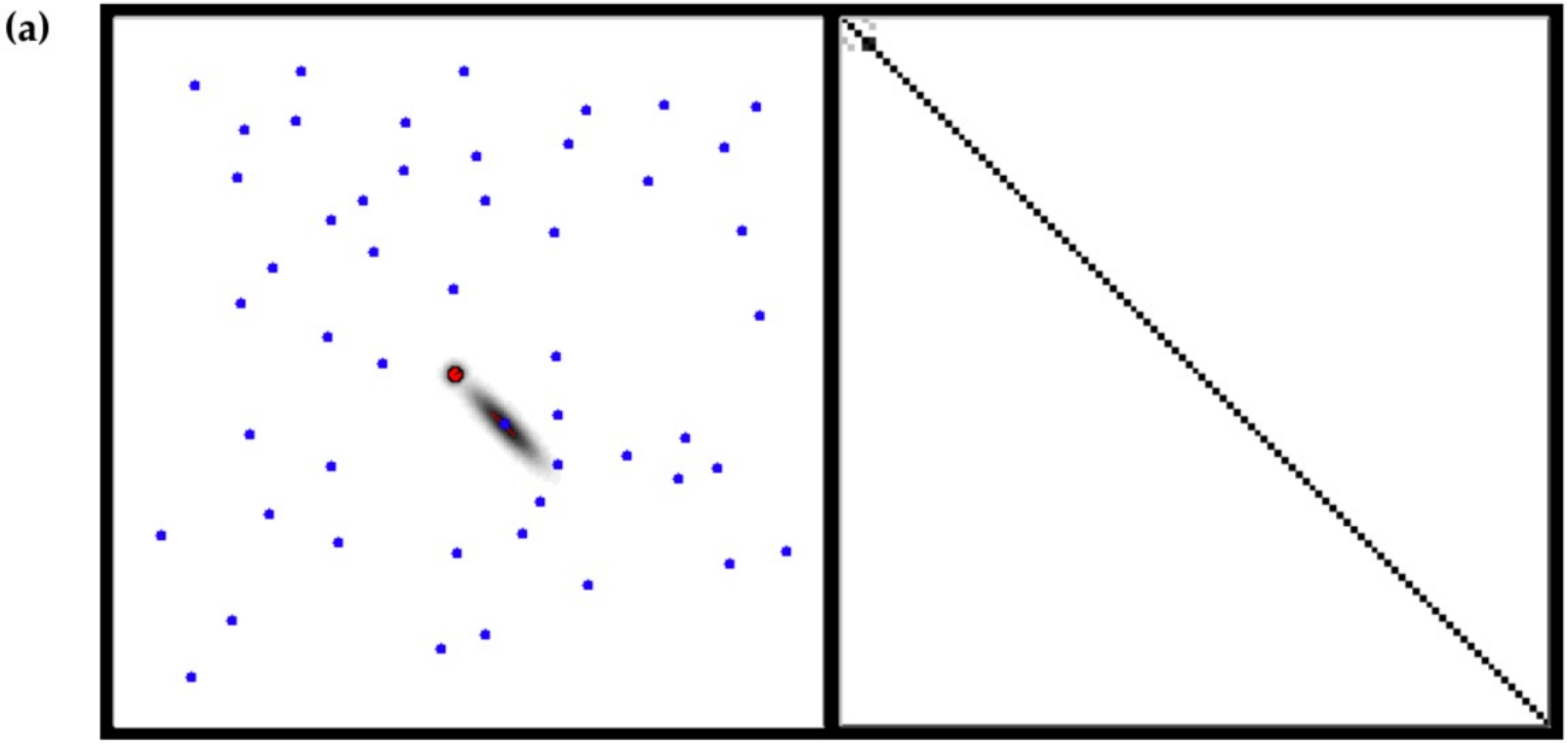
EKF SLAM – Illustration



EKF SLAM – Illustration



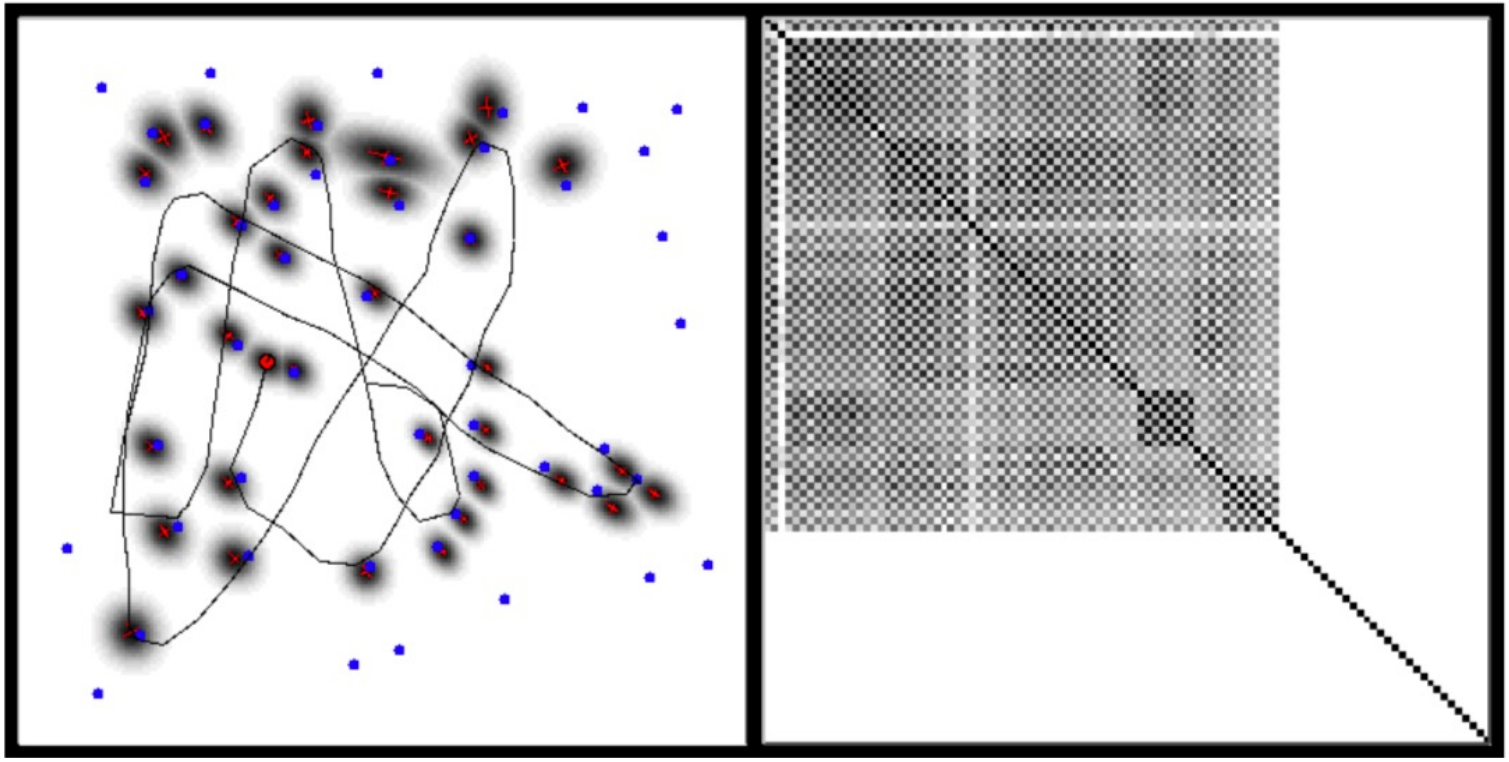
En simulation



Carte + trajectoire

Matrice de covariance

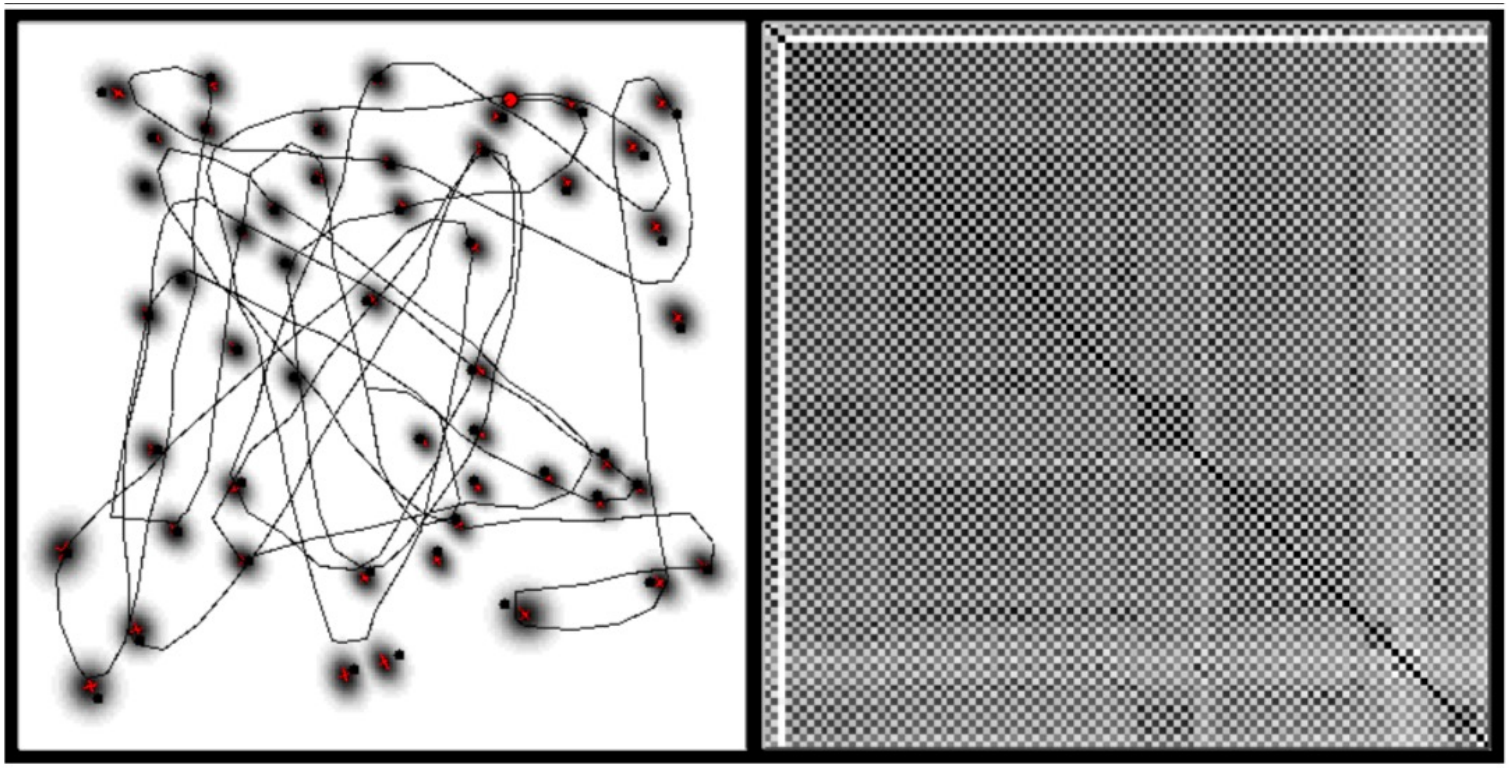
En simulation



Carte + trajectoire

Matrice de covariance

En simulation



Carte + trajectoire

Matrice de covariance

Covariance ’pleine’ à la fin

Retour en arrière ?

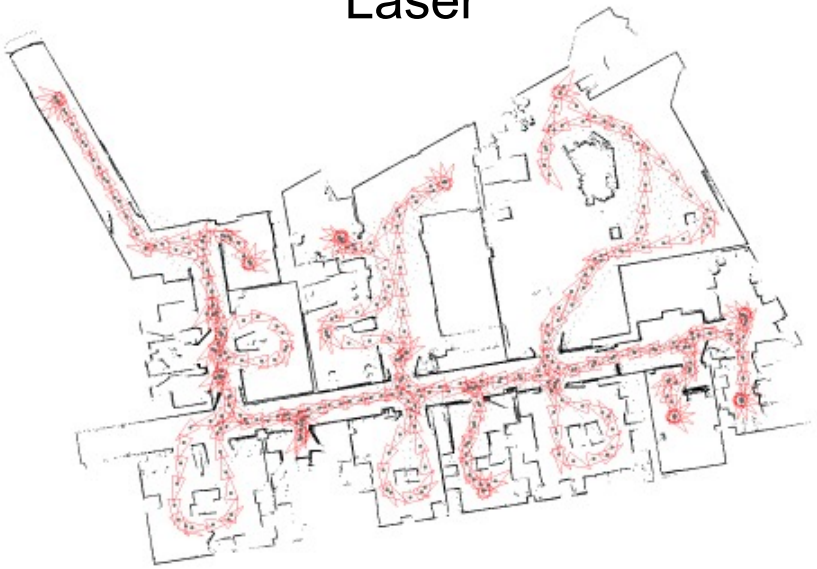
→ via la *matrice de covariances*

– Mémorise les relations entres éléments du vecteur d'états

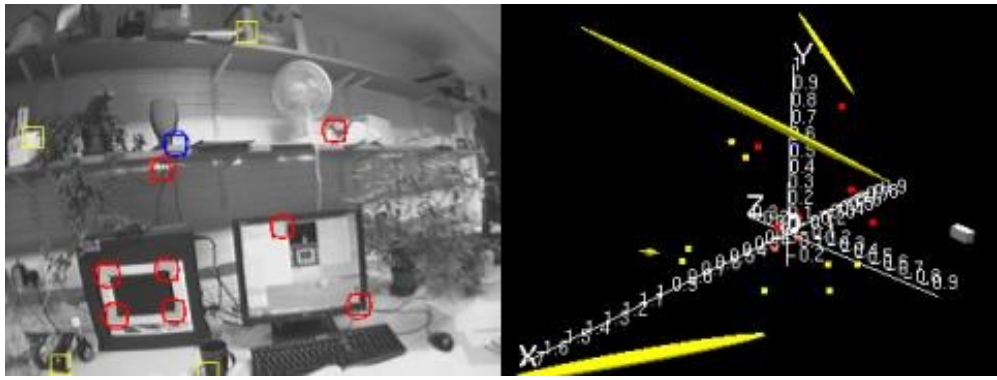
– Propage de nouvelles estimations aux éléments en relation

Implémentations

Laser



Vision



SLAM 3D avec une caméra monoculaire

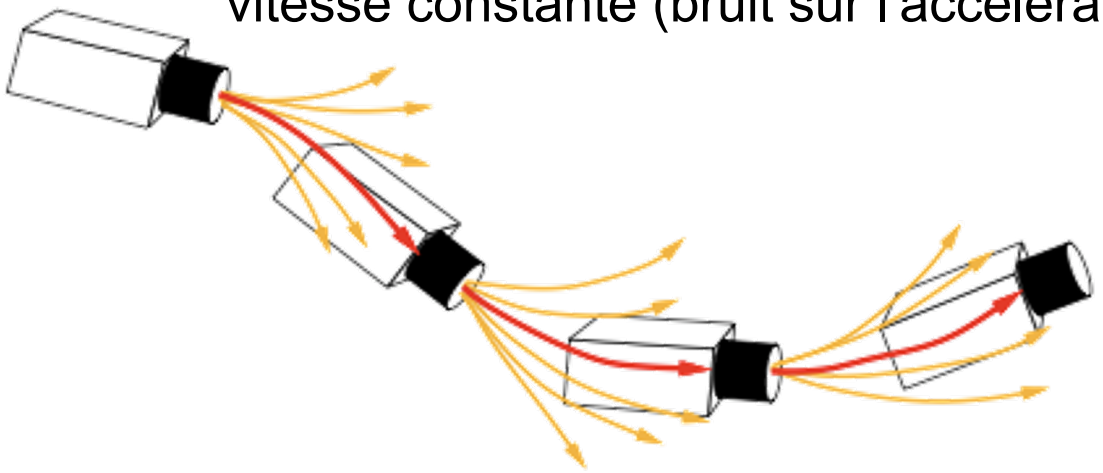
- *MonoSLAM: Real-Time Single Camera Davison*. Reid, Molton, and Stasse, PAMI 2007

- Problème de « structure from motion »
- Etat : position (6 ddl) + vitesses

$$\mathbf{x}_v = \begin{pmatrix} \mathbf{r}^W \\ \mathbf{q}^{WR} \\ \mathbf{v}^W \\ \omega^R \end{pmatrix}$$

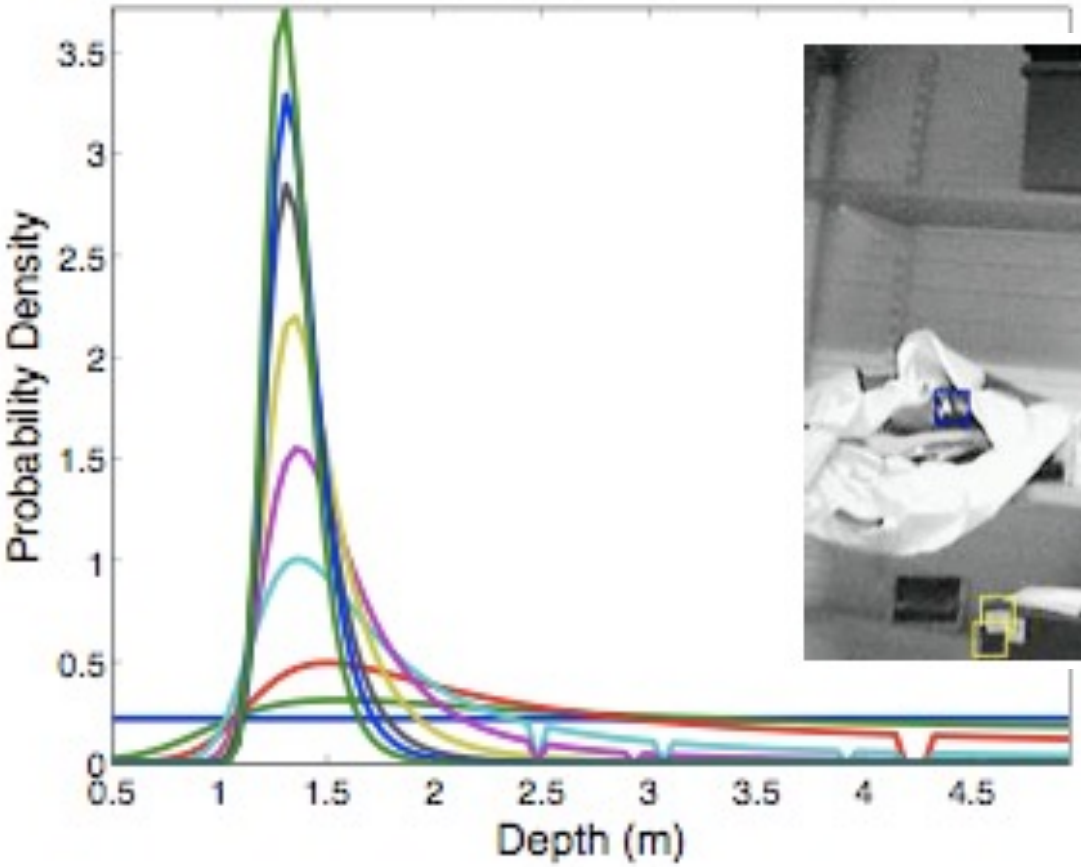
- Modèle de mouvement classique (Inertie, Odométrie) sur robot
- Modèle de mouvement sans odométrie (caméra seule)

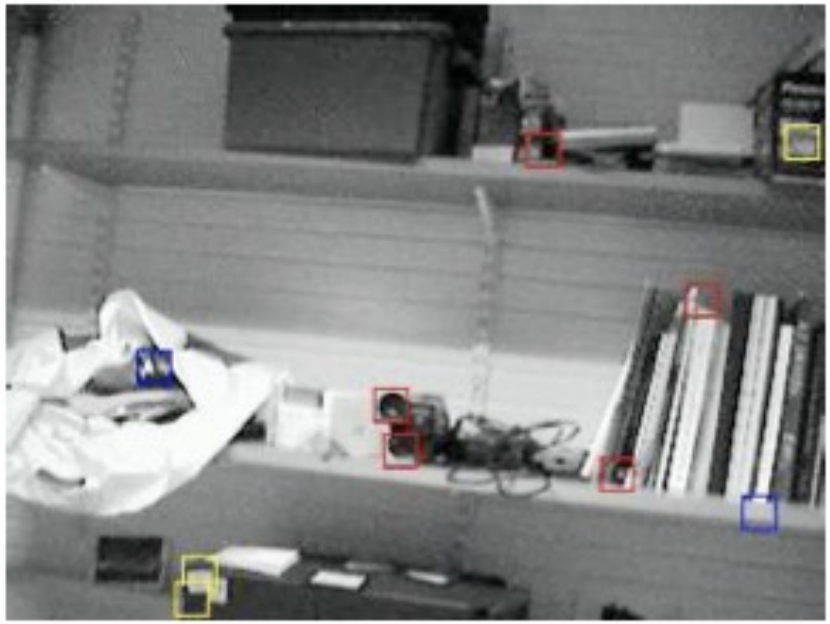
  vitesse constante (bruit sur l'accélération)



SLAM 3D avec une caméra monoculaire

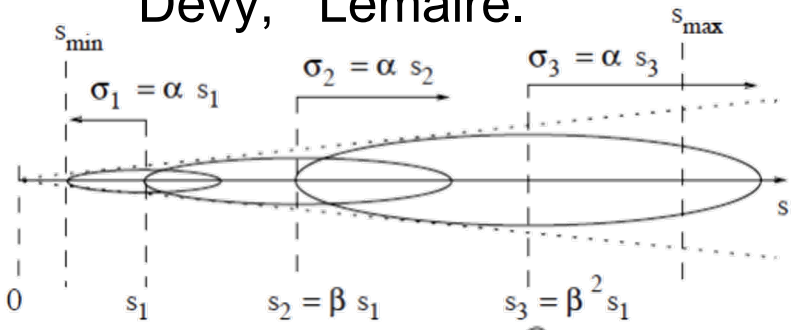
- Problème d'initialisation des features 3D d'après image
- Initialisation retardée : droite 3D + filtre particulaire auxiliaire pour distance

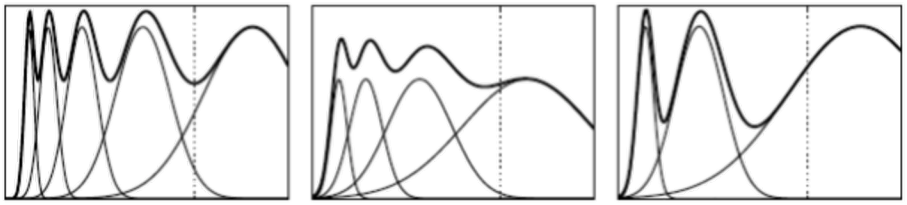


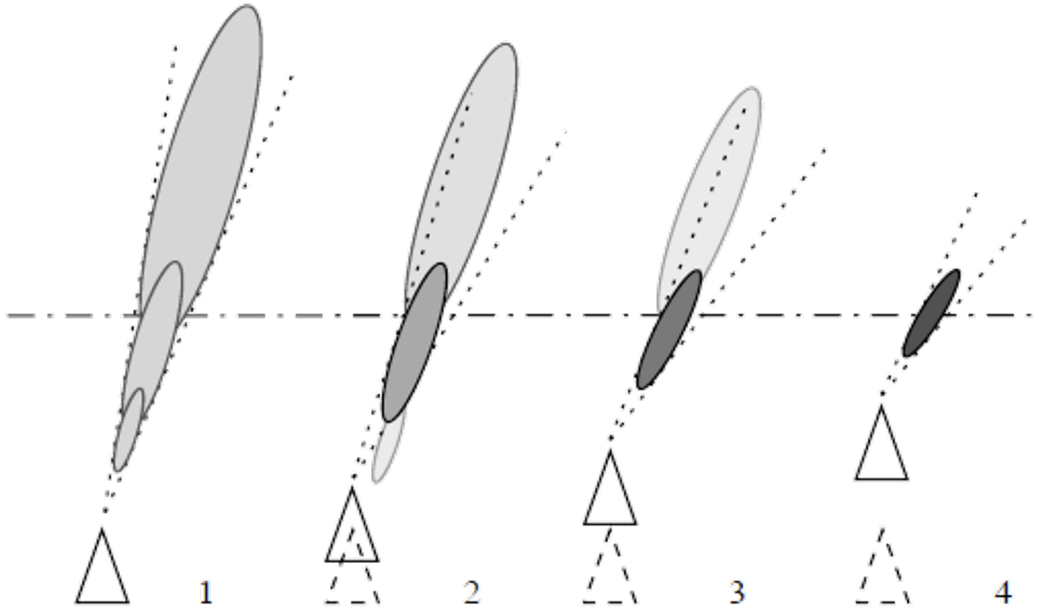


SLAM 3D avec une caméra monoculaire

- Initialisation non retardée : cone ≈ multi gaussiennes
- *Undelayed initialization in bearing only SLAM* Sola, Monin, Devy, Lemaire.

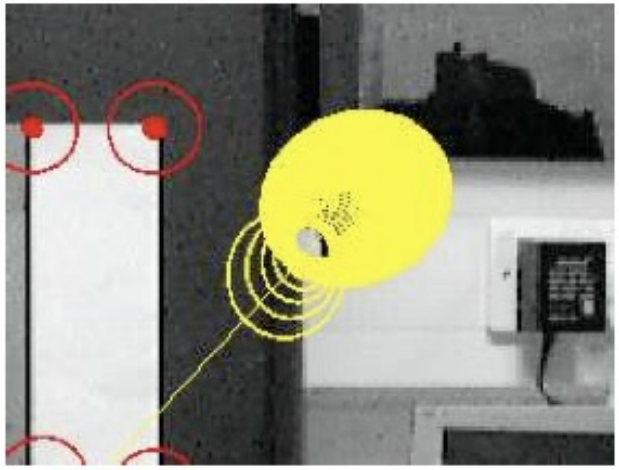




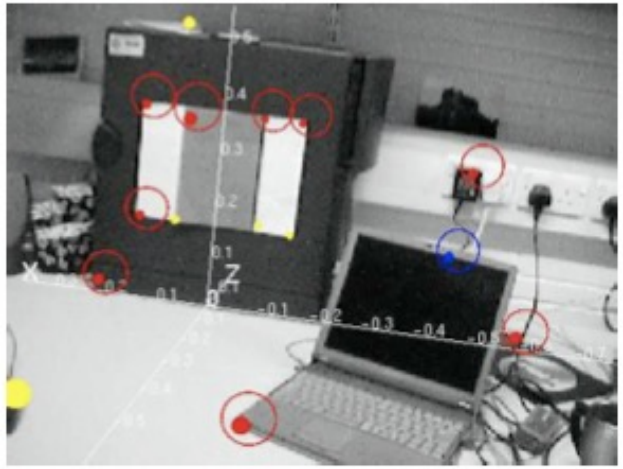


Association de données

- Correspondance carte <-> image
- Recherche active dans l'image



nouvelle feature
(filtre particulaire)



features de la carte

Mise à jour du filtre

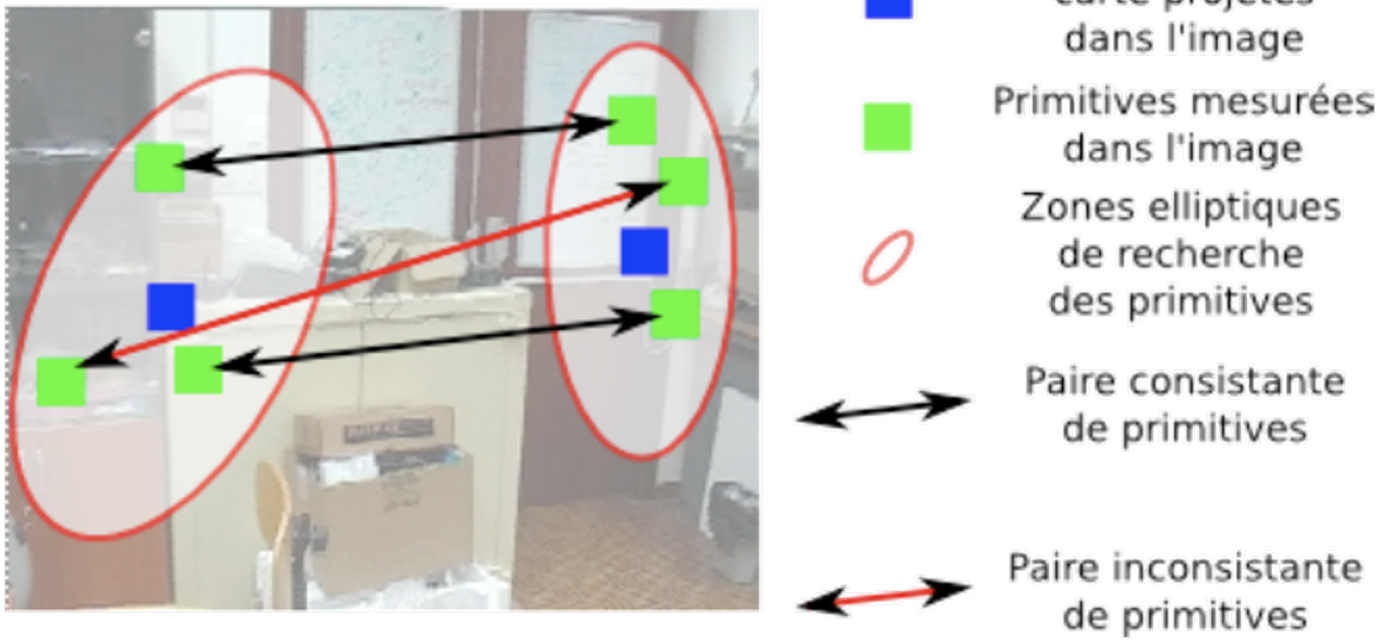
- Fonctionne à 30 Hz
- Carte limitée à 100 amers (EKF)

Exemple

Appariements

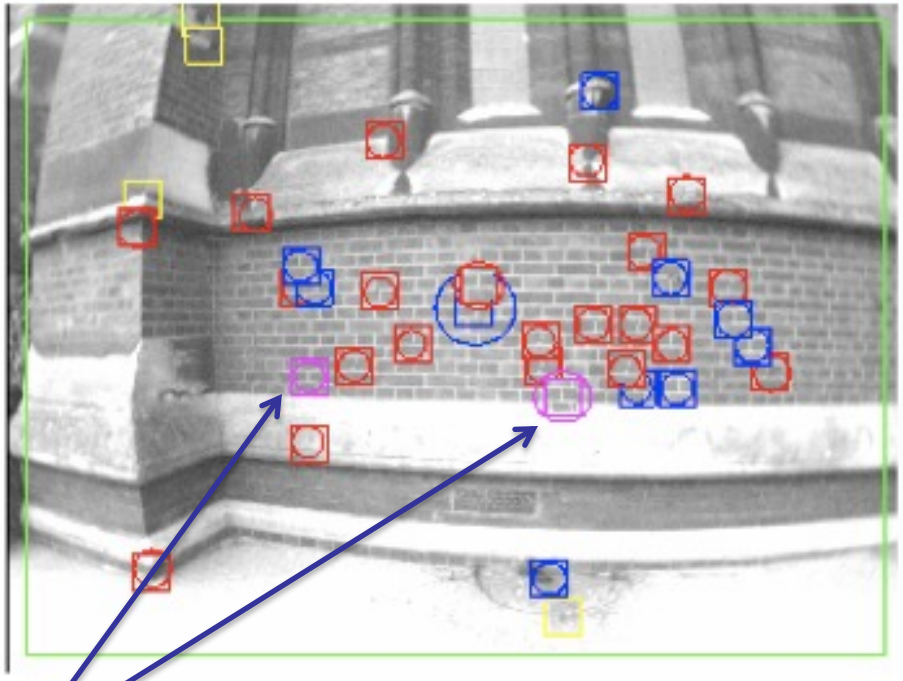
- Très sensible aux erreurs d'appariement amer perçu/amer de la carte.
- Une erreur peut entraîner un échec complet (dérive)

  -> Méthodes de matching plus robustes
- Joint Compatibility Test : Mahalanobis sur l'innovation conjointe



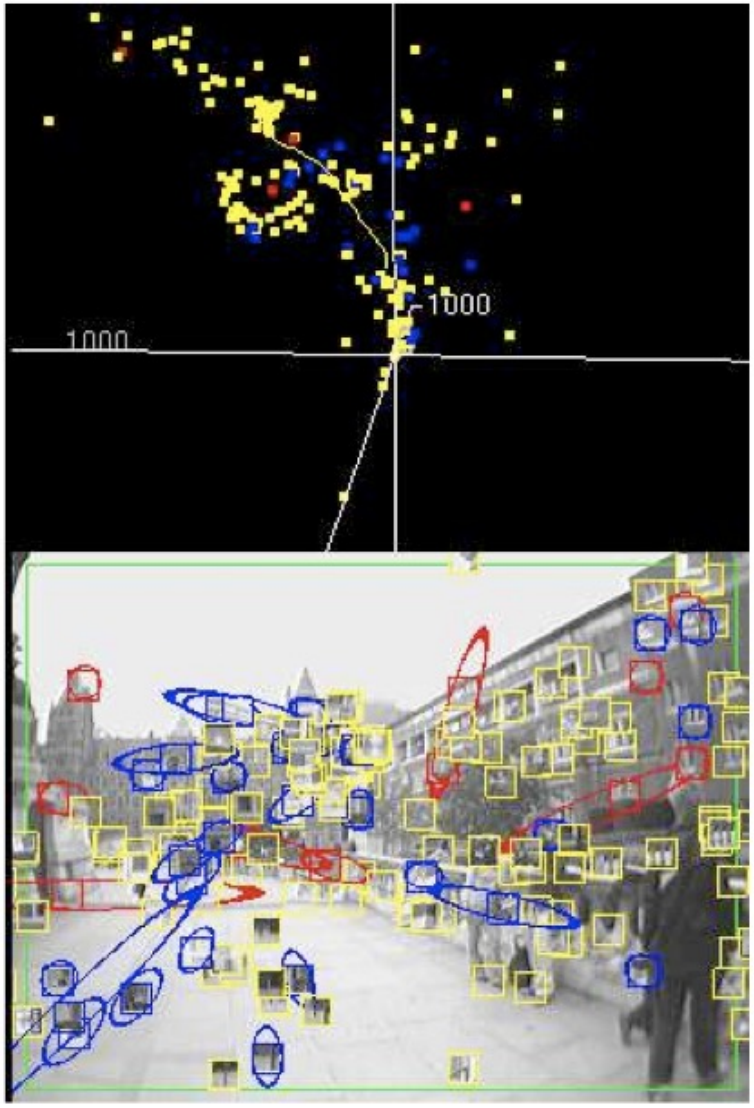
Joint Compatibility Test

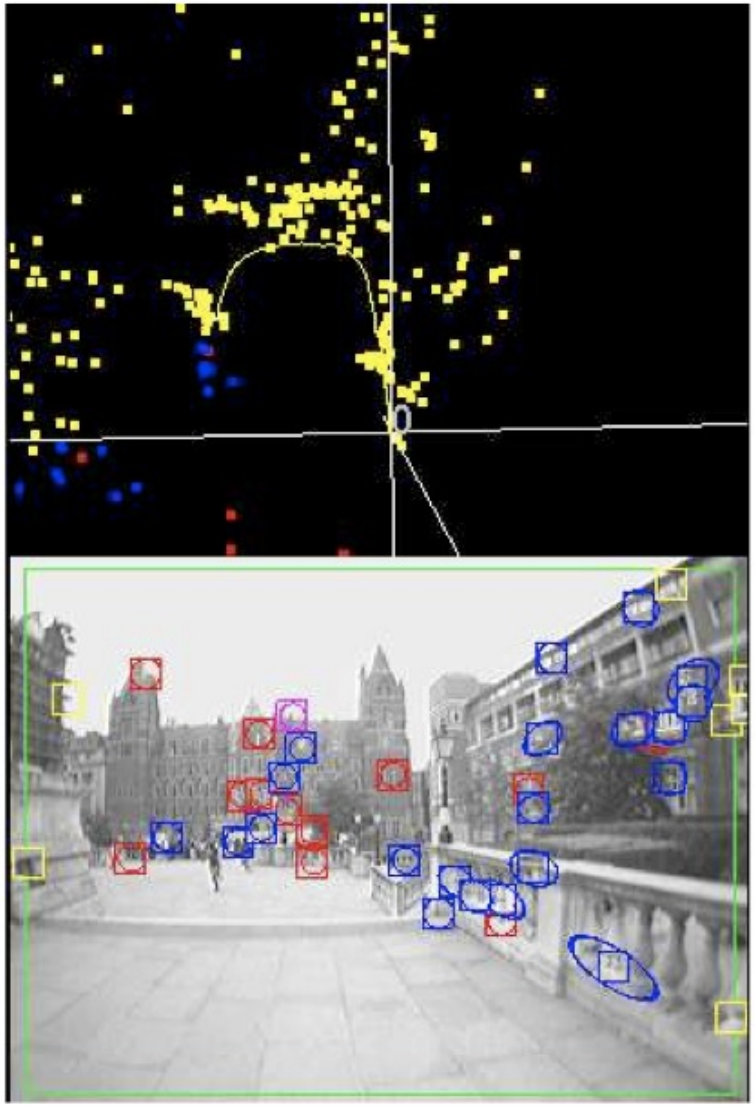
- Permet l'appariement de features ambigues
- Evite le matching de features mobiles



Rejeté

Joint Compatibility Test



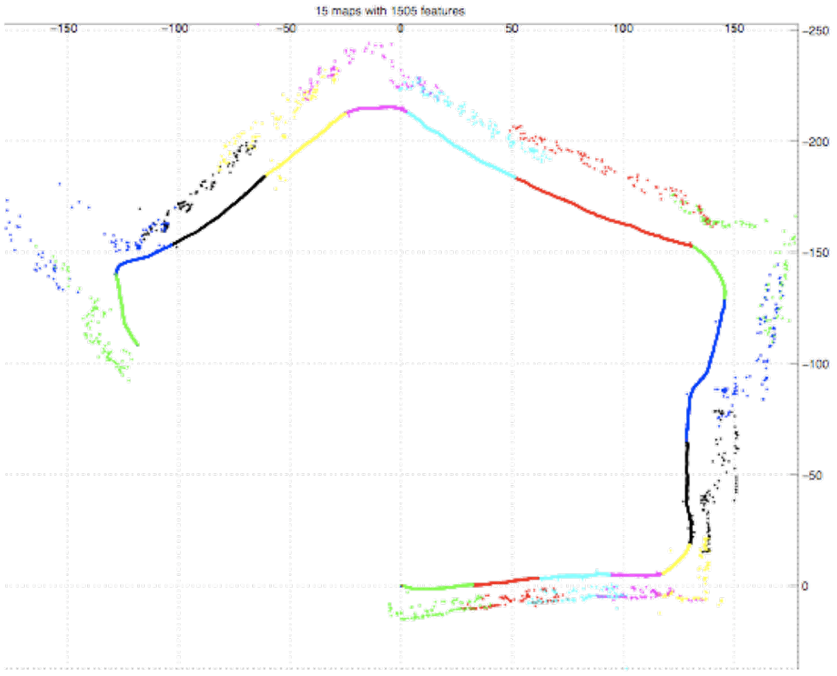


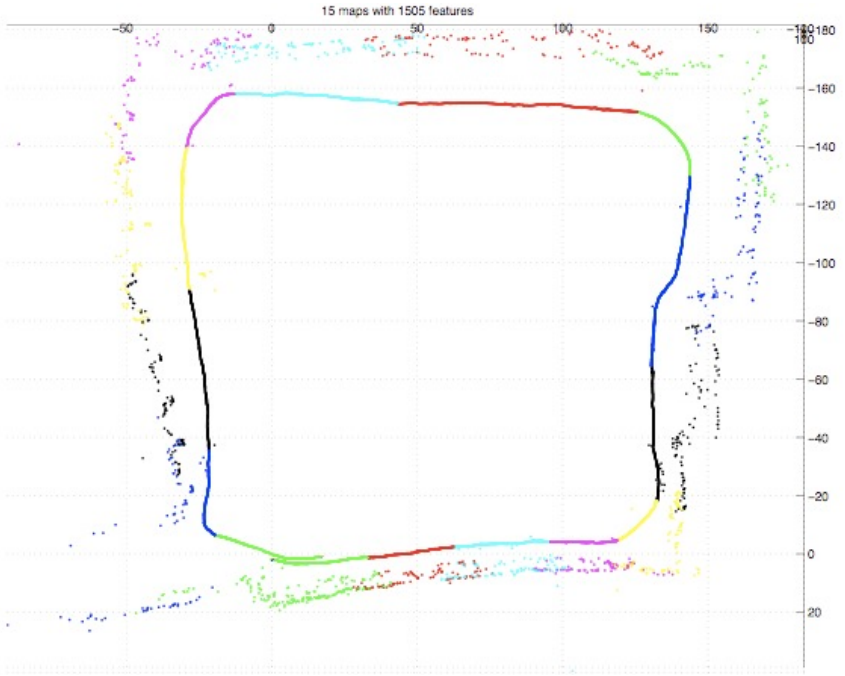
Complexité du SLAM EKF

- Complexité en O(N²) (matrice de covariance)
  - Négliger les covariances ne fonctionne pas
- Méthodes hiérarchiques
  - Cartes locales
  - Relations entre carte
- Création de sous-cartes
  - Nombre de features > seuil
  - Covariance des features > seuil
  - Structure de l'environnement

Utilisation de cartes locales

- Utilisation de cartes locales de taille M et calcul des covariances dans ces carte → O(M²)
- Détection de fermeture de boucle (map matching) + optimisation
- *Mapping Large Loops with a Single Hand-Held Camera* Laura A. Clemente, Andrew J. Davison, Ian Reid, José Neira and Juan D. Tardós, RSS 2007.





SLAM 01 - En résumé

- La cartographie est beaucoup plus complexe que la localisation, mais utilise les mêmes principes algorithmiques (filtrage, optimisation …)

- Un problème important est la dérive en découvrant l'environnement, apparent lors de la **fermeture de boucle,** qui est ignorée par les algorithmes de cartographie incrémentaux, mais corrigée par les algorithmes avec retour arrière

- En utilisant un laser et une grille d'occupation, des algorithmes incrémentaux tels que **Hector SLAM** sont robustes et précis

- Le **Filtre de Kalman Etendu** peut être utilisé pour la cartographie avec retour en arrière en intégrant directement la carte dans le vecteur d'état