IN 103 - Introduction à Matlab

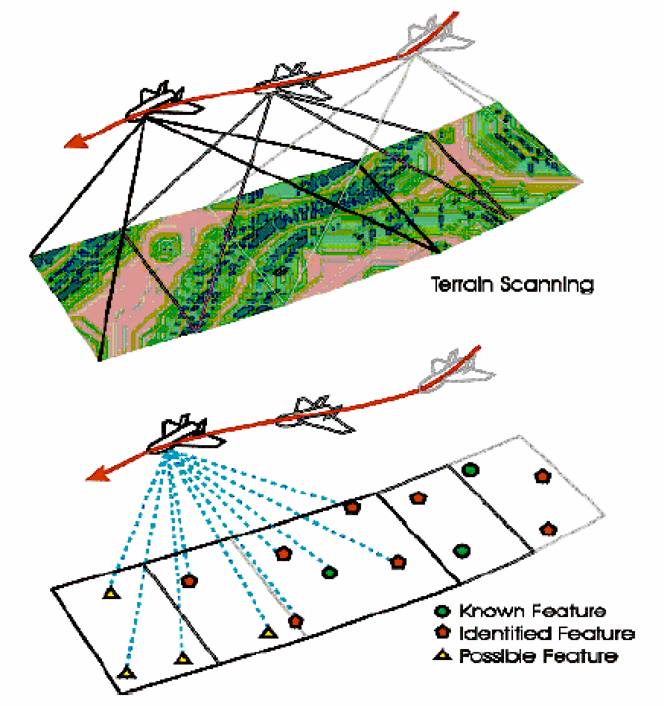
Projet robotique

Filtre de Kalman - Cartographie et localisation simultanées

David FILLIAT - david.filliat@ensta.fr

Construction d'une carte d'un environnement inconnu

- Permet au robot d'estimer sa position
- Fournit une information haut niveau
  → comportement planifié
- Permet à un utilisateur de connaître l'environnement



« Carte d'amers »

Cartographie

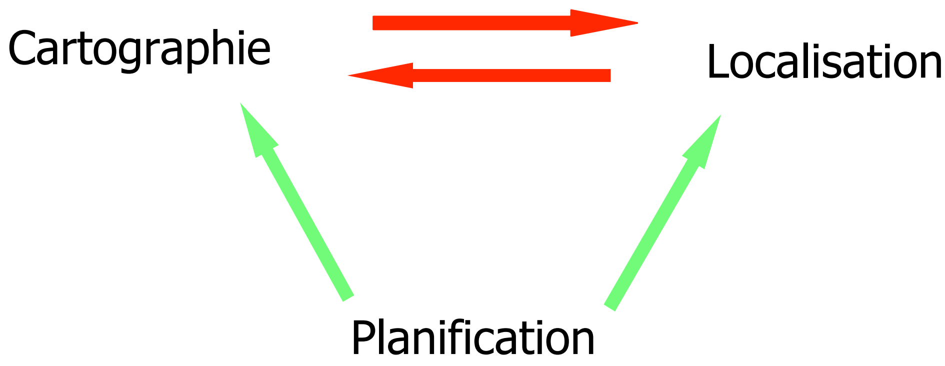
– Construction de la carte

Localisation

– Estimer la position du robot dans une carte connue

Planification

– Calculer un chemin de la position courante jusqu'au but



Robot autonome : Simultaneous Localization and Mapping (***SLAM***)

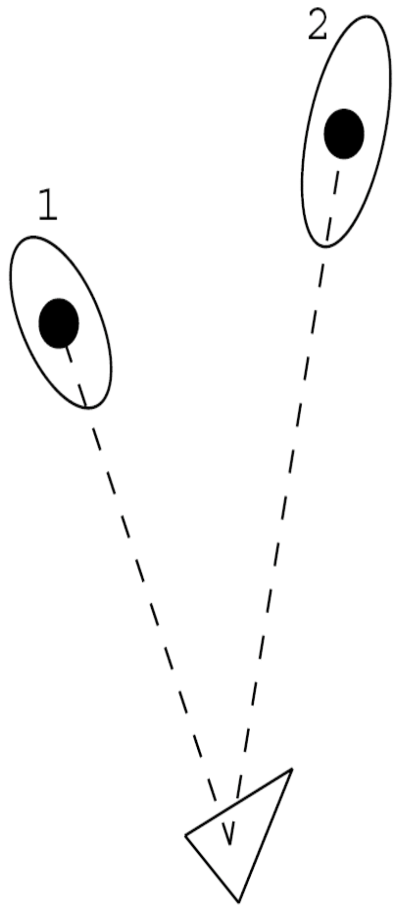
2 types d'informations aux propriétés complémentaires

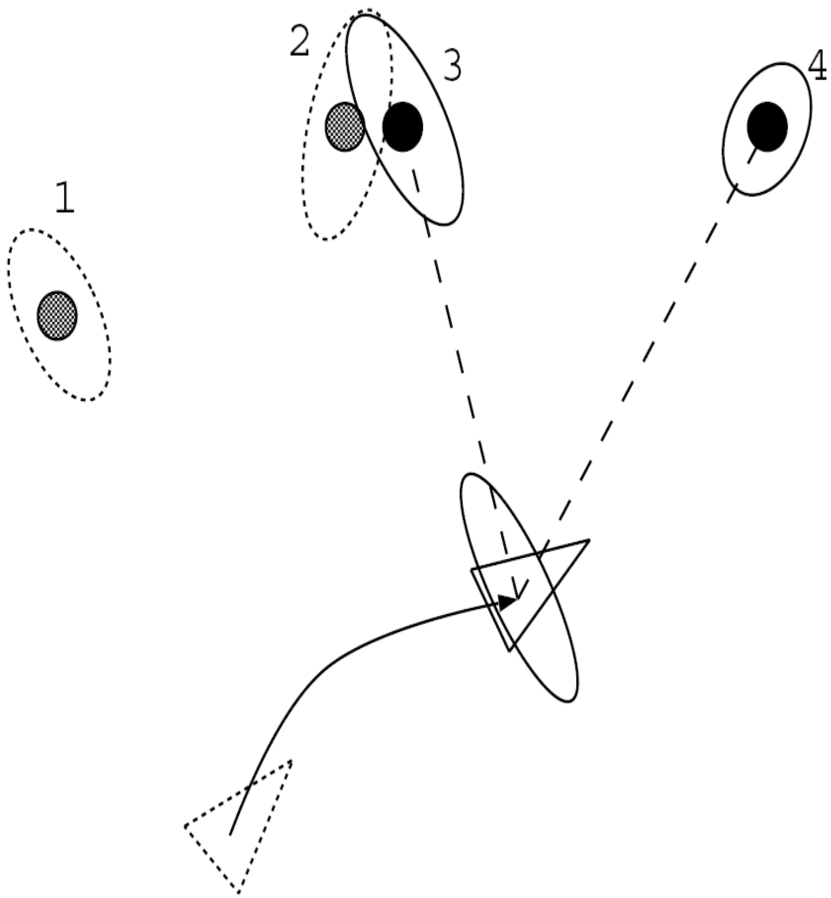
- Informations **internes** : informations *proprioceptives* (ou idiothétiques), renseignent sur les déplacements
  - Ex : Odométrie, inertie
  - **Erreur cumulative** (processus d'intégration)
  - Inutilisable à long terme
  - **Référence simple à utiliser, peu dépendante de la position**
- Informations **externes** : informations extéroceptives (ou allothétiques, *perceptions*), renseignent sur la position
  - Ex : Capteurs de contact, télémètre, caméra
  - **Erreur non cumulative**, mais :
    - **Perceptual aliasing**
    - **Variabilité perceptuelle**
  - Difficilement utilisables seules

➔ **Fusion des informations**

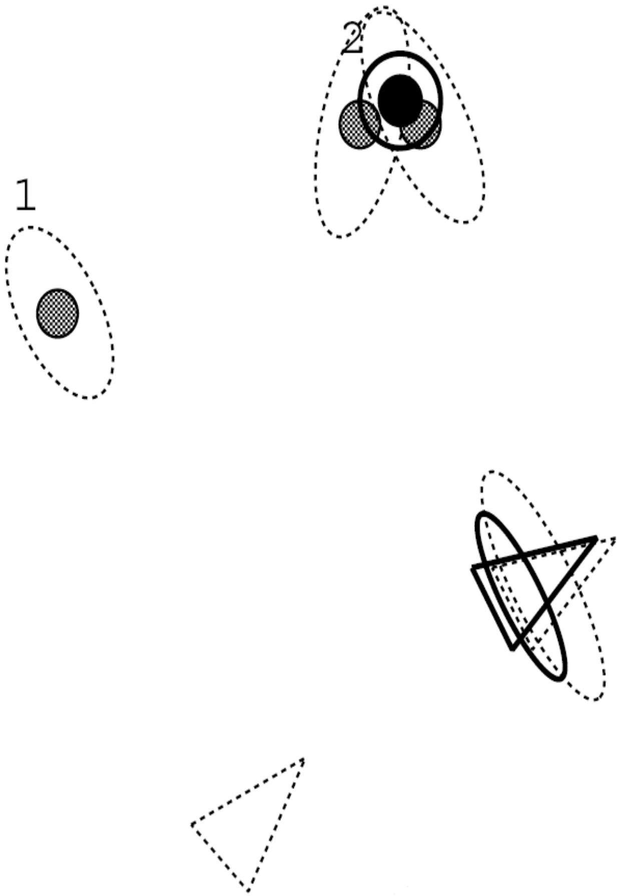
David FILLIAT - david.filliat@ensta.fr

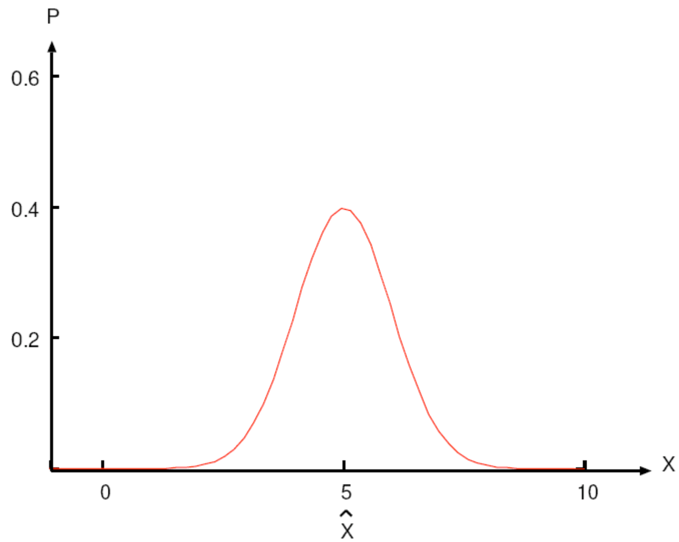
SLAM - Illustration 1




2
3
4
1

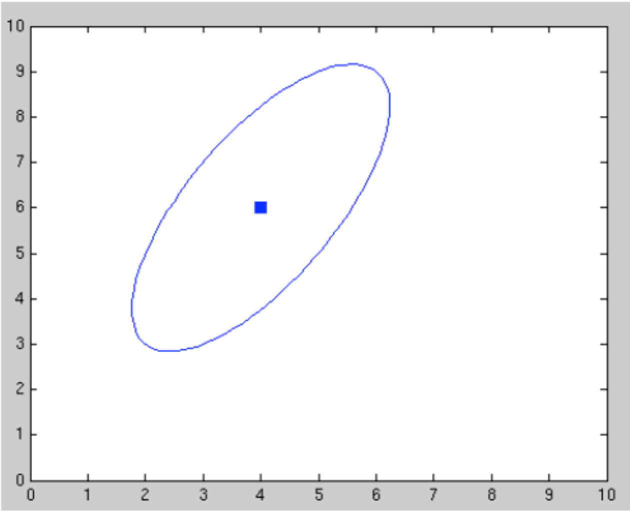
SLAM - Illustration 3





Loi gaussienne

$$p = \frac{1}{\sigma\sqrt{2\pi}} exp(-\frac{(x-\hat{x})^2}{2\sigma^2}$$



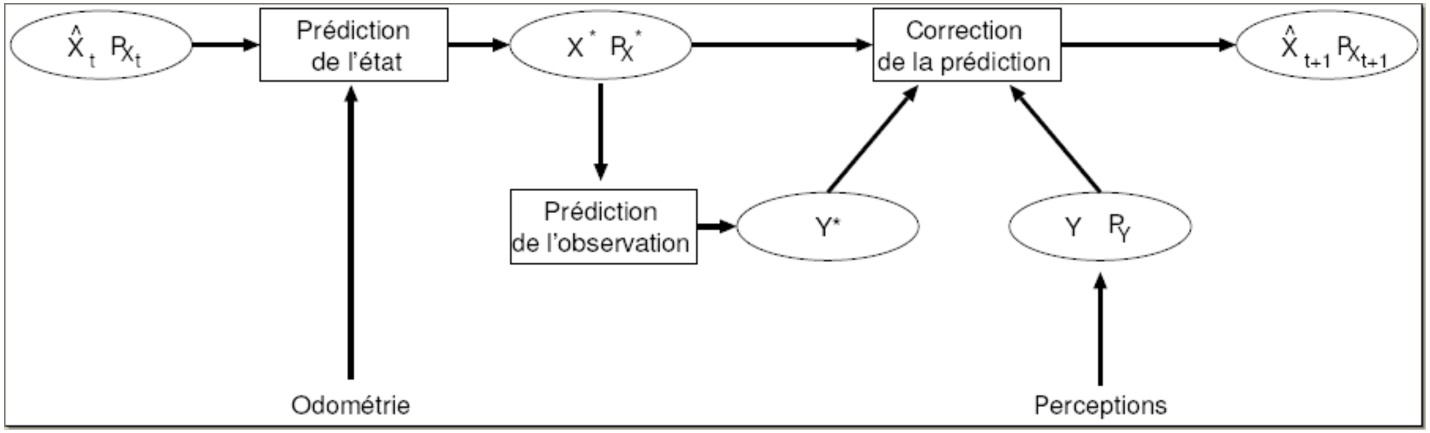
$$\hat{X} = \frac{1}{N}\sum_{1}^{N} X_i$$

$$\Sigma^{a,b} = \frac{1}{N}\sum_{1}^{N}(X_i^a - \widehat{X^a}) * (X_i^b - \widehat{X^b})$$

➔ **Représentée et manipulée par sa moyenne et variance**

On cherche à estimer la position du robot et la carte à partir de leur évolution et de mesures :

Filtrage de Kalman



$$X_t = \begin{bmatrix} x \\ y \\ x_{a1} \\ y_{a1} \\ \vdots \\ x_{aN} \\ y_{aN} \end{bmatrix}$$

Etat : position du robot + carte (position des amers)

Evolution : déplacement du robot

Perception : position relative robot-amers

Incertitude (gaussienne) sur l'estimation : matrice de covariance P

4 étapes :

1 Prédiction de l'état : $X_t^* = A.\hat{X}_{t-1} + B.u_t$ $\quad P_t^* = A.\hat{P}_{t-1}.A^T + Q$

Par ex u = mesure de l'odométrie

2 Prédiction de l'observation pour l'état prédit : $Y_t^* = H.X_t^*$

Par ex : mesure de la position du robot (H=I) par triangulation en utilisant la carte

3 Observation sur le système réel : Y

4 Correction de la prédiction :

$$\hat{X}_t = X_t^* + K(Y - Y_t^*)$$

$$\hat{P}_t = P_t^* - KHP_t^*$$

Avec le gain :

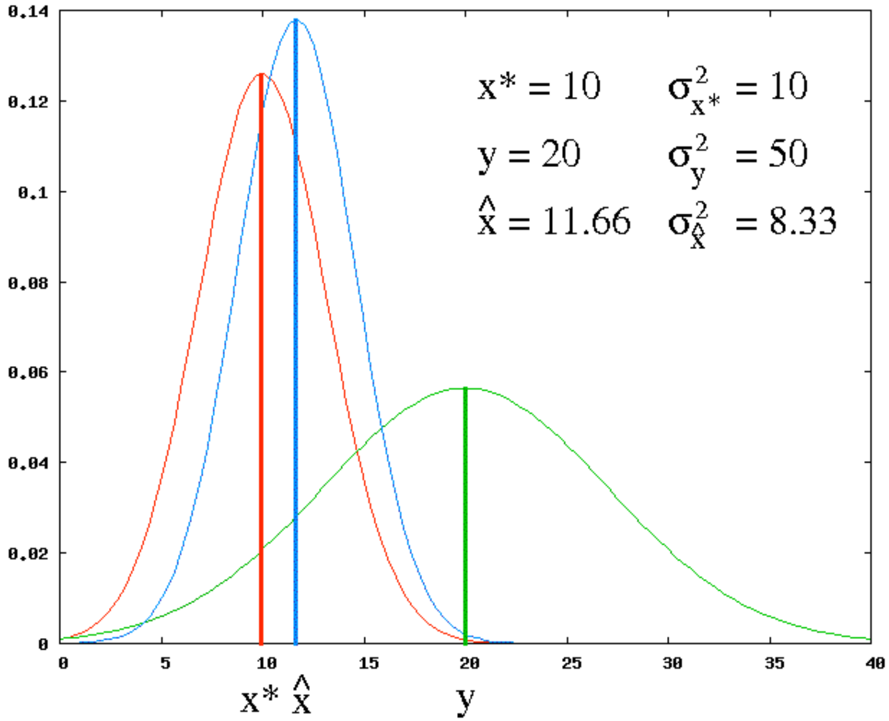
$$K = P_t^* H^T.(H.P_t^*.H^T + P_Y)^{-1}$$

Illustration en dimension 1 :

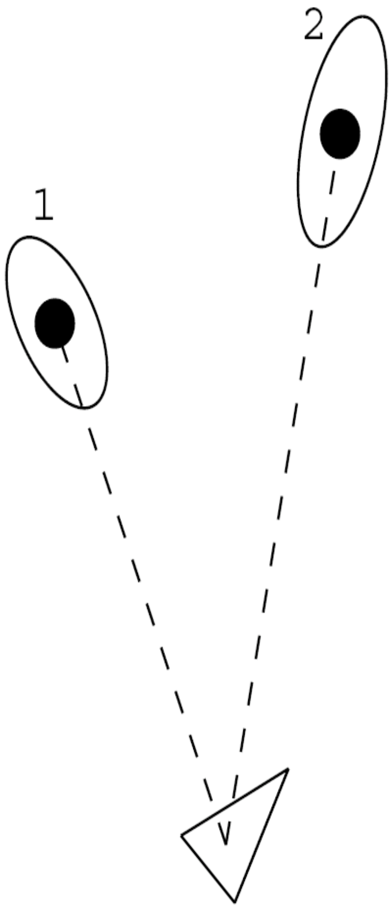
$$K = \frac{\sigma_x^{*2}}{\sigma_x^{*2} + \sigma_y{}^2}$$

$$\hat{x} = x^* + \frac{\sigma_x^{*2}}{\sigma_x^{*2} + \sigma_y{}^2}(y - x^*)$$

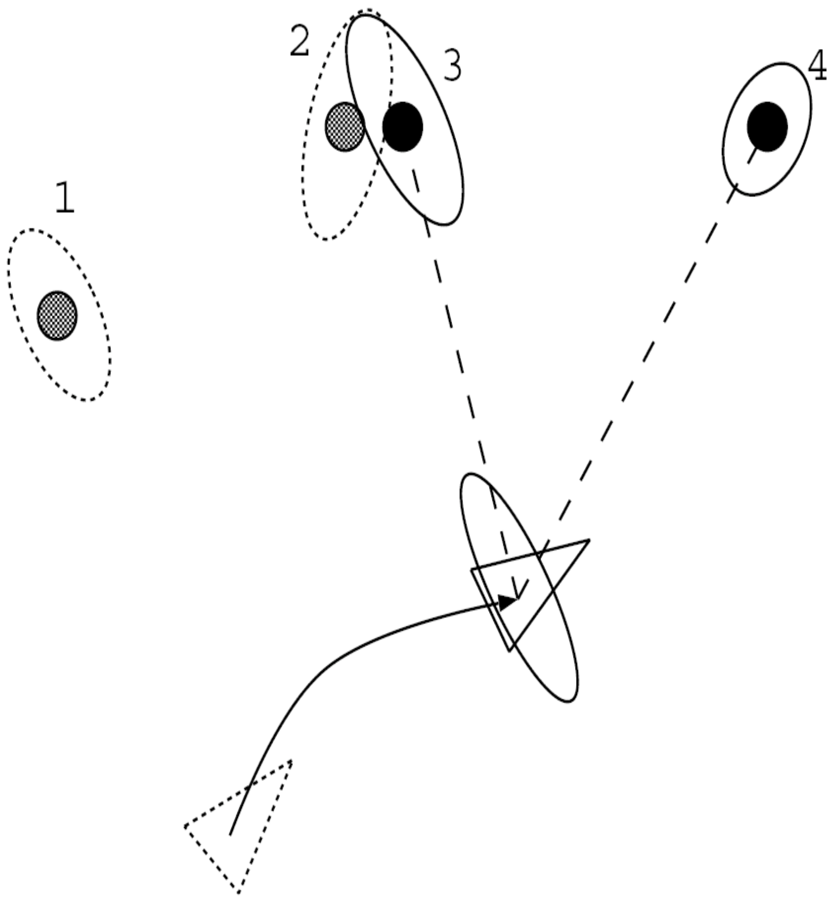
$$= \frac{\sigma_x^{*2} y + \sigma_y{}^2 x}{\sigma_x^{*2} + \sigma_y{}^2}$$



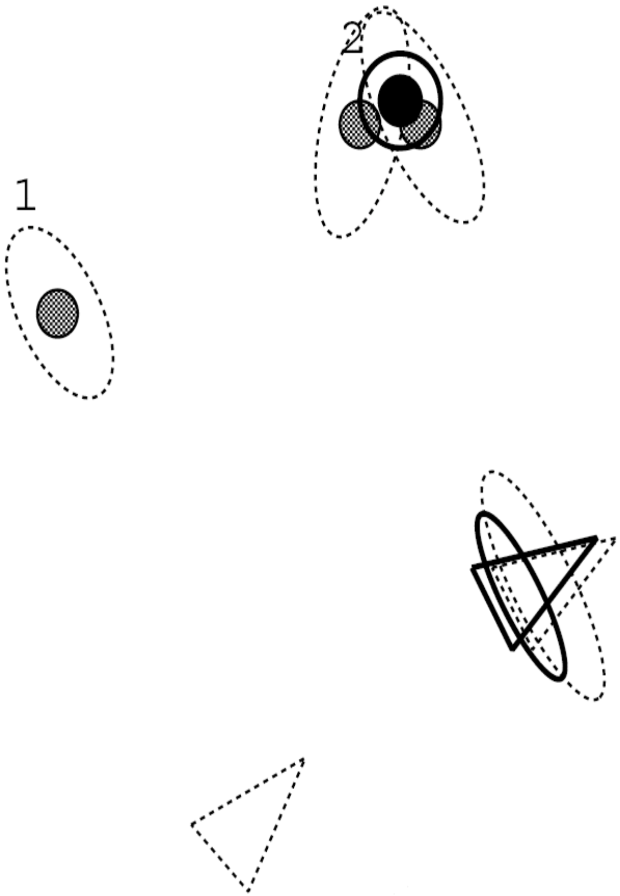
SLAM - Illustration 1



SLAM - Illustration 2



SLAM - Illustration 3



Traitement 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
  - Si il n'est pas dans la carte : ajout au vecteur d'état et à la matrice de covariance
- Pour savoir s'il est dans la carte :
  - Perception d'amers uniques
  - En cas d'incertitude (Perceptual aliasing):

Choix de l'observation la plus proche de la prédiction

- en distance euclidienne
- ou en distance de mahalanobis *(pondération par l'incertitude)*

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

Pour les problèmes non linéaires (cas de la localisation en robotique)

– Filtre de Kalman étendu

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

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

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

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

Algorithm 5 Algorithme de SLAM pour un déplacement u et un amer perçu Y

Prédiction de l'état $X_t^* = f(\hat{X}_{t-1}, u_t)$
Estimation de la position et de la variance de l'amer perçu $x_Y, y_Y, \sigma_{x_Y}, \sigma_{y_Y}$
for all Amer i **do**
  Calcul de la distance de Mahalanobis $d_i^2 = \frac{1}{2}(Y - Y_i)^T (P_Y + P_{Y_i})^{-1} (Y - Y_i)$
end for
Sélection de l'amer j de la carte tel que d_j soit minimal
if $d_j <$ Seuil : l'amer est déjà dans la carte **then**
  Prédiction de l'observation $Y_{j_t}^* = h_j(X_t^*)$
  Correction de l'état prédit $\hat{X}_t = X_t^* + K(Y - Y_{j_t}^*)$
else
  ajout de l'amer perçu à la carte
end if

Application possible en vision

Andrew Davison : http://www.doc.ic.ac.uk/~ajd/Movies/kitchen.mp4.avi