Localisation par filtrage de Kalman

David FILLIAT

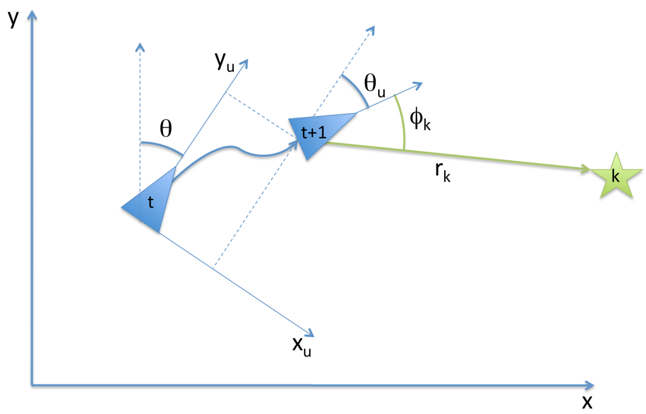
4 décembre 2009

1 Introduction

Dans ce TD, nous allons appliquer le filtrage de Kalman étendu à la localisation d'un robot. Pour cela, nous utiliserons le code MATLAB réalisé par Paul Newman (Site du cours) légèrement modifié, que vous téléchargerez sur ma page.

Ces programmes permettent de simuler un robot se déplaçant sur une trajectoire donnée dans un environnement constitué d'amers ponctuels. Ils implémentent une méthode de filtrage de Kalman étendu simple utilisant la perception de la direction et de la distance de ces amers ponctuels.

2 Modèles



Le robot modélisé se déplace sur un plan et perçoit la direction et la distance d'amers ponctuels situés sur ce même plan. Son état est représenté par sa position et son orientation dans un repère global :

$$X_t = [x_t, y_t, \theta_t]^T$$

Le déplacement du robot entre les instants t et $t+1$ est mesuré grâce à l'odomérie donnée dans le repère du robot à l'instant t :

$$U_t = [x_u, y_u, \theta_u]^T$$

Le modèle d'évolution de l'état est donc donné par :

$$X_{t+1} = f(X_t, U_t) = \begin{bmatrix} x_t + x_u cos(\theta_t) - y_u sin(\theta_t) \\ y_t + x_u sin(\theta_t) + y_u cos(\theta_t) \\ \theta_t + \theta_u \end{bmatrix}$$

avec un bruit gaussien donné par une matrice de covariance Q.

Les perceptions fournissent des mesures de la distance et de la direction d'un amer k supposé parfaitement identifiable :

$$Z_t = \left[r_t^k, \phi_t^k \right]^T$$

Le modèle d'observation correspondant est donc :

$$Z_t = h^k(X_t) = \begin{bmatrix} \sqrt{(x_k - x_t)^2 + (y_k - y_t)^2} \\ atan2(\frac{y_k - y_t}{x_k - x_t}) - \theta_t \end{bmatrix}$$

ou x_k et y_k sont les coordonnées (connues) de l'amer dans le repère global. Ce modèle est entaché d'un bruit gaussien de matrice de covariance P_Y.

3 Filtre de Kalman étendu

Les modèles étant non linéaires, nous allons utiliser le filtre de Kalman étendu afin d'intégrer les mesures d'odométrie et les perceptions des amers pour produite une estimation $\hat{X}_t$ la position du robot X_t. Rappelons les équations du filtre de Kalman étendu :

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

ou les matrices A, B et H sont les jacobiennes des fonctions f et h :

$$A_{ij} = \frac{\partial f_i(x,u)}{\partial x_j}$$

$$B_{ij} = \frac{\partial f_i(x,u)}{\partial u_j}$$

$$H_{ij} = \frac{\partial h_i(x)}{\partial x_j}$$

Le code Matlab fournit réalise ce filtrage dans une carte d'amers tirée aléatoirement pour un robot se déplaçant sur une trajectoire sinusoïdale dans cette carte[1].

4 Question 1 - Jacobiennes

Calculez les jacobiennes A, B et H et écrivez les fonctions correspondantes dans le code Matlab (fonctions GetObsJac(xPred, iFeature,Map), A(x,u) et B(x,u) à la fin du fichier EKFLocalisation.m). Exécutez ensuite la fonction *EKFLocalisation* et observez les résultats produits. Que représentent les trois courbes affichées à la fin du code ?

[1] Le code est incomplet et ne fonctionnera pas en l'état.

5 Question 2 - Panne de capteur

Modifiez la fonction GetObservation(k) pour simuler une panne de capteur, i.e., renvoyer $z = []$ durant un certain laps de temps. Que se passe-t-il au niveau du filtre de Kalman pendant ce temps ?

6 Question 3 - Erreurs de modèle

Les matrices de covariances utilisées dans le filtre (QEst, PYEst) reflètent fidèlement le bruit des capteurs du robot (QTrue et PYTrue qui sont utilisées pour le simulateur). Etudiez le comportement du filtre lorsque le bruit utilisé par le filtre est grossièrement sur ou sous-estimé. Que se passe-t-il en particulier lorsque le bruit utilisé sur les perceptions est très faible ? Que se passe-t-il lorsque le bruit estimé sur la distance des amers est grand et celui sur la direction des amers est correct ?

7 Question 4 - Observabilité partielle

L'un des aspects intéressants du filtre de Kalman est la possibilité de n'avoir que des observations partielles. C'est déjà le cas avec la version du filtre implantée (l'état est de dimension 3 alors que les observations sont de dimension 2). Il est cependant possible d'utiliser encore moins d'information, par exemple de n'utiliser que la distance ou que la direction des amers. Modifiez le modèle et le code Matlab (fonctions GetObservation, DoObservationModel, matrice PY etc...) pour n'utiliser que la distance des amers. Que peut-on observer sur les performances du filtre ?