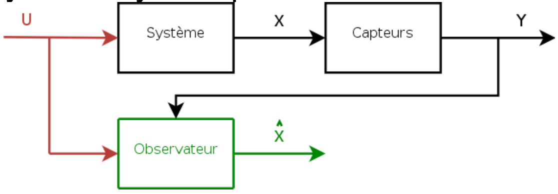
TP FUSION DE DONNÉES PAR FILTRAGE DE KALMAN

Introduction

En automatique et en traitement du signal, lorsque l'état d'un système n'est pas mesurable, on implémente un **observateur** qui permet de reconstruire l'état à partir d'un modèle du système dynamique et des mesures d'autres grandeurs.



La théorie de l'observateur d'état déterministe a été introduite dans les années soixante par Luenberger pour les systèmes linéaires. Kalman a également formulé un observateur en considérant un système linéaire stochastique. Pour les systèmes non-linéaires, l'observation reste un domaine où la recherche est très active, mais l'utilisation la plus commune est l'emploi d'un filtrage de Kalman étendu (EKF). Le filtre de Kalman estime les états d'un système dynamique à partir d'une série de mesures incomplètes ou bruitées. Il s'agit d'un estimateur récursif. Cela signifie que pour estimer l'état courant, seul l'état précédent et les mesures actuelles sont nécessaires. L'historique des observations et des estimations n'est pas requis.

Filtre de Kalman classique (systèmes linéaires)

Les équations d'état d'un système linéaire s'écrivent
$$x_k = Fx_{k-1} + Bu_k + w_k$$
$$y_k = Hx_k + v_k$$

Avec :

- k l'itération courante
- F,B les matrices qui relient respectivement l'état précèdent et la commande à l'état courant x_k
- w_k le bruit du processus, caractérisé par sa matrice de covariance Q
- H la matrice qui relie l'état courant à la mesure y_k
- v_k le bruit de mesure, caractérisé par sa matrice de covariance R

La sortie du filtre est représentée par deux variables :

- $\hat{x}_{k|k}$ l'estimation de l'état à l'instant k;
- $P_{k|k}$ matrice de covariance de l'erreur (mesure de la précision de l'état estimé).

Puisque c'est un filtre récursif, il doit avoir en entrée les valeurs initiales $\hat{x}_{1|1}$ et $P_{1|1}$
Le filtre de Kalman a deux phases distinctes: **prédiction** et **mise à jour**. La phase de prédiction utilise l'état estimé de l'instant précédent pour produire une estimation de l'état courant. Dans l'étape de mise à jour, les observations de l'instant courant sont utilisées pour corriger l'état prédit afin d'obtenir une estimation plus précise.

Prédiction
$$\hat{x}_{k|k-1} = F\hat{x}_{k-1|k-1} + Bu_k$$
$$P_{k|k-1} = FP_{k-1|k-1}F^T + Q$$

Mise à jour

$$\tilde{y}_k = y_k - H\hat{x}_{k|k-1} \quad \text{innovation}$$
$$S_k = HP_{k|k-1}H^T + R \quad \text{covariance de l'innovation}$$
$$K_k = P_{k|k-1}H^T S_k^{-1} \quad \text{gain de Kalman optimal}$$
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k\tilde{y}_k \quad \text{mise a jour de l'etat}$$
$$P_{k|k} = (I - K_k H_k)P_{k|k-1} \quad \text{mise a jour de la covariance}$$

Filtre de Kalman étendu (pour systèmes non-linéaires)

Les équations d’état d’un système linéaire s’écrivent : $\begin{array}{l} x_k = f(x_{k-1}, u_k, w_k) \\ y_k = h(x_k + v_k) \end{array}$

Avec f et h des fonctions différentiables. La fonction f peut être utilisée pour calculer l'état prédit à partir de l'état estimé précédent et la fonction h pour calculer l'observation prédite à partir de l'état prédit. Cependant, f et h ne peuvent pas être appliqués directement au calcul de la covariance : une matrice des dérivées partielles, la **Jacobienne**, est calculée. À chaque instant, les Jacobiennes de f et h sont évaluées avec les états estimés courants et employées dans les équations du filtre. Ce processus linéarise essentiellement la fonction non linéaire autour de l'estimation courante. On obtient les équations du filtre de Kalman étendu suivantes :

Prédiction

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_k, 0)$$
$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q$$

Mise à jour

$$\tilde{y}_k = y_k - h(\hat{x}_{k|k-1}, 0) \quad \text{innovation}$$
$$S_k = H_k P_{k|k-1} H_k^T + R \quad \text{covariance de l'innovation}$$
$$K_k = P_{k|k-1} H_k^T S_k^{-1} \quad \text{gain de Kalman optimal}$$
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k\tilde{y}_k \quad \text{mise a jour de l'etat}$$
$$P_{k|k} = (I - K_k H_k)P_{k|k-1} \quad \text{mise a jour de la covariance}$$

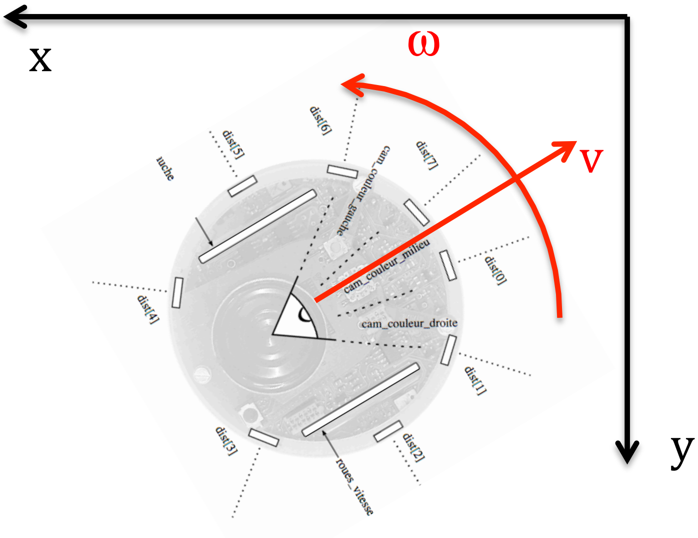
Où les matrices de transition et d'observation sont définies comme étant les Jacobiennes suivantes : $\begin{array}{l} F_k = \left.\dfrac{\partial f}{\partial x}\right|_{\hat{x}_{k|k-1}, u_k} \\ H_k = \left.\dfrac{\partial h}{\partial x}\right|_{\hat{x}_{k|k-1}} \end{array}$ Remarque : la convergence de ce filtre n'est aucunement assurée car il s'agit d'une convergence locale. En fait, il existe de nombreux exemples pour lesquels la convergence dépend de l'initialisation de l'état.

Objectif du TP

Il s’agit de fusionner des données de vitesse et de position permettant d’estimer la position d’un véhicule (**e-puck**). Pour réaliser cette expérience, vous devez d’abord génerer deux jeux de données, tous deux issus d’une manipulation avec le **e-puck**.

Plateforme

Vous disposez d'une machine ubuntu et d'un robot e-puck connecté à la machine via wifi. On considère le e-puck comme un véhicule non holonome idéal (ne glissant pas, avec des roues infiniment fines) décrivant une trajectoire pré-définie dans le repère (x, y) ayant pour origine l'intersection des parois.



Sur l'e-puck, on exploite :

- les mesures des **codeurs des roues**, pour obtenir les vitesses instantanées axiale et angulaire (v, ω) et variations correspondantes de x, y :

$$\begin{cases} \dot{\theta} = \omega \\ \dot{x} = v\cos\theta \\ \dot{y} = v\sin\theta \end{cases}$$

  Ces mesures, sont sauvegardées dans les fichiers: **v.txt dx.txt dy.txt**
- les capteurs **infrarouges de distance** pour estimer (en détectant les parois de l'environnement) les position x et y du robot dans le repère monde. Ces mesures sont sauvegardées dans : **x.txt y.txt**

Campagne de mesures:

- Aller dans le fichier du programme client : **cd soft/HMEE351_epuck**
- Supprimer les anciens fichiers de mesure (tous les .txt sauf CMakeLists.txt)
- Laisser le sélecteur de mode du epuck sur A
- Allumer le e-puck sur le coté
- Attendre l'activation de la wifi (dongle wifi clignotant en bleu). Sinon, redémarrer le e-puck (et eventuellement changer la batterie)
- Vérifier la connexion : **ping 192.168.1.2**
- Positionner le e-puck sur la grille en position (x,y) = (0.18, 0.12)
- Lancer le programme : **./main c**
- Noter la position finale (x,y) du e-puck sur la grille
- Récupérer sur un clé USB les fichiers :
  x.txt y.txt dx.txt dy.txt v.txt t.txt
  pour traitement sous Matlab. La donnée temps **t.txt** représente la mesure de temps à chaque acquisition. On suppose que l'échantillonnage se fait à temps

constant. Les positions sont données en mètre, les vitesses en mètre par seconde et le temps en secondes.
- Passer les données sur une machine équipée de Matlab

Filtre linéaire

Vous utiliserez les fichiers : **`x.txt`** **`y.txt`** **`dx.txt`** **`dy.txt`** **`t.txt`**

1) Chargez les données de chaque expérimentation et visualisez les. Visualisez la trajectoire mesurée (x,y). Reconstruisez la trajectoire à l'aide des données de vitesse en utilisant : **`EstimateFromSpeeds.m`** Tirez-en des conclusions.

2) Ecrivez les équations d'évolution et de mesure, écrivez puis implantez votre filtre de Kalman, en utilisant : **`EstimateFromKalman.m`**.

On suppose que les mesures précises mais entachées de bruits aléatoires (centrés et blancs). On considère que les mesures de position sont entachées d'un bruit de variance 0.7 sur l'axe x et de 0.8 sur l'axe y, et que les mesures de vitesse sont aussi entachées de bruit de 0.3 sur l'axe x et 0.2 sur l'axe y.

La solution immédiate qui vous viendrait à l'esprit serait de modéliser complètement le robot mobile non-holonome. Ce n'est pas la solution la plus simple et surtout elle risque de vous mener vers des modèles non-linéaires (qui ne sont pas la spécialité des filtres de Kalman). Nous vous conseillons donc d'adopter la solution suivante : vous approximé le mouvement du véhicule en x par :

$$\begin{cases} x_{k+1} = x_k + \Delta t \ \dot{x}_k + \dfrac{\Delta t^2}{2} \ \ddot{x}_k \\ \dot{x}_{k+1} = \dot{x}_k + \Delta t \ \ddot{x}_k \end{cases}$$

(et idem pour y). Le robot mobile ne donnant pas des accélérations très fortes pendant sa trajectoire, celle ci peut être considérée comme une perturbation de variance σ_{acc} à spécifier. Cela donne, dans le filtre : $Q = TT^T\sigma_{acc}$

avec $T = \begin{bmatrix} \Delta t^2/2 & 0 \\ 0 & \Delta t^2/2 \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix}$ et on peut utiliser $\sigma_{acc} = 0.3$

N'oubliez pas de paramétrer votre solution de façon à pouvoir jouer sur les différents paramètres. Entre autre essais voyez ce qui se passe lorsque sont modifiés :

• l'initialisation du vecteur d'état,
• l'initialisation de la matrice de variance de l'état (Pk),
• la variance de l'accélération.

Filtre étendu

Vous utiliserez les fichiers : **`x.txt`** **`y.txt`** **`v.txt`** **`t.txt`**.

On ne dispose plus du vecteur vitesse dans les deux directions mais du vecteur d'avancée du robot (c'est a dire de la norme du vecteur vitesse). Cette donnée est entachée d'un bruit de variance 0.3.

3) Implantez votre filtre de Kalman, en utilisant : **`EstimateFromExtKalman.m`**

Si la relation entre la mesure et les observations dans la première expérience est triviale et linéaire, il n'en est pas de même dans la seconde expérience. Le plus simple est de supposer que la vitesse d'avancée v_k est reliée à $\dot{x}_k$ et $\dot{y}_k$ par :

$$v_k = \sqrt{(\dot{x}_k)^2 + (\dot{y}_k)^2}$$