Evaluation : Observateurs et filtre de Kalman

- **Tout document autorisé**, mais **communication** avec d'autres personnes **interdite**.
- **Rendre code MATLAB** et autres documents informatiques sur **Moodle** dans un **dossier zip** et éventuellement des documents manuscrits complémentaires si nécessaire. S'assurer que les documents rendus sont bien vos versions finales. Vérifier avec l'encadrant que les fichiers sont bien rendus.
- **Mettre en commentaires** directement dans le code toutes les informations et **explications** nécessaires pour que le correcteur comprenne ce qui est fait.
- **Ne pas rester bloqué si le code ne s'exécute pas** correctement, s'efforcer d'apporter une réponse (éventuellement partielle) à toutes les questions.
- Eventuellement partir des codes des TD précédents pour aller plus vite.
- Fournir une **capture d'écran** (appuyer sur la touche Print Screen/Impression Ecran et faire coller dans Microsoft Paint) des divers résultats intermédiaires importants.
- **Faites des sauvegardes régulières.** Surveillez régulièrement si votre PC doit redémarrer.
- 25 points sont disponibles dont 5 qui seront du bonus, la note finale étant sur 20.

Part 1 : Questions théoriques (durée estimée : 20 min)

1. Quelle utilité voyez-vous au filtre de Kalman ? [**2 points**]
2. A quoi servent les matrices de covariance dans le filtre de Kalman ? [**2 points**]
3. On distingue souvent 2 grandes étapes dans le filtre de Kalman (même si parfois seule l'une d'entre elle est utilisable selon les conditions). Lesquelles ? [**1 point**]

Part 2 : Filtre de Kalman et régulation d'un robot vélo autonome (durée estimée : 130 min)

Pour améliorer le confort d'utilisation des Vélib, il pourrait être intéressant que ceux-ci soient autonomes et capables de revenir tous seuls à leur station d'attache après usage. Pour y arriver, ce robot vélo électrique serait équipé d'un actionneur (u1) réglant l'angle du guidon, d'un moteur (u2) contrôlé en vitesse pour avancer, d'une boussole pour connaitre son cap (theta) et d'un GPS pour connaitre sa position x,y.

Le problème d'équilibre du vélo est supposé déjà résolu indépendamment.

La boussole sera considérée comme suffisamment précise donc theta pourra être considéré comme connu, au même titre que les entrées u1 et u2. Par contre, en pratique le GPS en zone

urbaine avec grands immeubles et rues étroites est souvent très bruité (« sauts » de position, etc.). Pour pouvoir se localiser quand même précisément, nous pouvons utiliser un modèle d'état simple et le fusionner avec les données GPS à l'aide d'un filtre de Kalman.

Un simulateur simple contrôlable au clavier est disponible ici : http://www.ensta-bretagne.fr/lebars/Share/velo_simu.zip .

1. D'après la description et le simulateur proposé, quelles sont les entrées du robot ? Quel est le vecteur d'état ? Quelle est l'équation d'évolution ? **(Durée estimée : 10 min)** [**3 points**]

2. Linéariser l'équation d'évolution du robot pour qu'elle devienne compatible avec les hypothèses du filtre de Kalman : donner Ak, uk et indiquez si vous passez par un vecteur d'état intermédiaire (souvent noté zk) **(Durée estimée : 20 min)** [**3 points**].

3. La position initiale sera considérée comme assez bien connue car en général, l'opérateur sait assez précisément où il a démarré son robot. Rajouter le code nécessaire pour indiquer que l'on connait la position initiale du robot avec une variance de 0.1 (environ 0.5 m d'erreur avec une probabilité de 99%) **(Durée estimée : 10 min)** [**1 point**].

4. Supposons que l'on n'ait pas de GPS. Ainsi, nous n'avons pas de mesures directes de x,y, donc nous n'avons pas de bruit de mesure non plus. Cependant, notre modèle étant forcément imprécis, nous avons du bruit d'état : on peut considérer par exemple qu'il est de variance 0.001 sur x et y (à ajuster en fonction des tailles des ellipses qui seront dessinées par la suite). En fonction de toutes ces informations, utiliser la fonction kalman pour estimer à chaque instant la position du robot avec sa matrice de covariance associée **(Durée estimée : 30 min)** [**4 points**].

5. Dessiner les ellipsoïdes de confiance contenant la position du robot avec une probabilité de 90% **(Durée estimée : 10 min)** [**2 points**].

6. Modifier ensuite le code pour rajouter la prise en compte du GPS. Pour un GPS à bas coût, on peut considérer que le bruit de mesure est avec une variance de 2 (environ 5 m d'erreur avec une probabilité de 99%) **(Durée estimée : 15 min)** [**2 points**].

7. Rajouter le suivi de waypoints (si votre code avec le Kalman, ne fonctionne pas, vous pouvez juste reprendre pour base le simulateur de robot vélo http://www.ensta-bretagne.fr/lebars/Share/velo_simu.zip et supposer qu'on a le droit d'accéder directement à toutes les variables d'état) **(Durée estimée : 20 min)** [**5 points bonus**].