



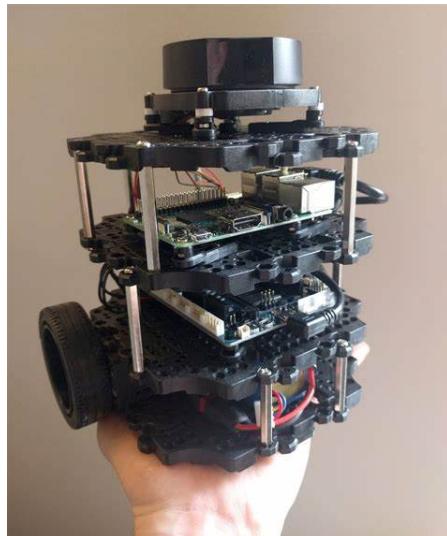
POLYTECHNIQUE
MONTRÉAL

WORLD-CLASS
ENGINEERING

POLYTECHNIQUE MONTRÉAL

ELE6705 : TRAITEMENT NUMÉRIQUE DES SIGNAUX

Projet



BOXUS FLORENT
2411548

OUSMAN DIAWARA 2321537

PROFESSEUR : HASSAN
BENSALAH

Année académique : 2024-2025

Table des matières

1	Introduction	2
2	Mise En Contexte	2
2.1	Modèle de déplacement	3
2.2	Modèle d'observation	3
3	Modèle d'état	4
4	Modèle d'état étendu	4
5	EKF	5
6	EFIR	7
7	Conclusion	11

1 Introduction

Les applications de robots mobiles nécessitent souvent une localisation automatique rapide, précise et peu coûteuse. Bien que ce problème ait été résolu par différentes méthodes depuis des décennies, la triangulation traditionnelle est encore utilisée dans de nombreux cas, en s'appuyant sur des informations provenant de trois nœuds ayant des coordonnées connues. Ces méthodes peuvent impliquer des nœuds stationnaires actifs avec un récepteur rotatif ou des repères passifs avec un émetteur-récepteur rotatif. Cependant, ces méthodes souffrent souvent d'une précision insuffisante dans des environnements bruyants, nécessitant des estimateurs optimaux. La théorie de l'estimation propose plusieurs méthodes, dont le filtre de Kalman étendu (EKF), qui est largement utilisé en robotique. Bien qu'efficace, l'EKF présente des limites, telles que des estimations biaisées et une sensibilité élevée au bruit. D'autres approches, comme le filtre de Kalman non linéaire (UKF), le modèle de Markov caché (HMM) et le filtre de particules (PF), ont été développées pour surmonter ces problèmes. Un autre filtre alternatif, le filtre EFIR, ignore totalement les statistiques de bruit et d'erreur initiale et nécessite encore des investigations pour la localisation robotique. Dans le cadre de ce projet, nous étudierons la méthode de triangulation à l'aide du filtre de kalman étendu et de l'EFIR.

2 Mise En Contexte

Pour réaliser l'analyse, imaginons un robot comme dans l'article [1], qui se déplace dans un plan en 2 dimensions, triangulé par 3 beacons tel qu'indiqué sur le schéma 1.

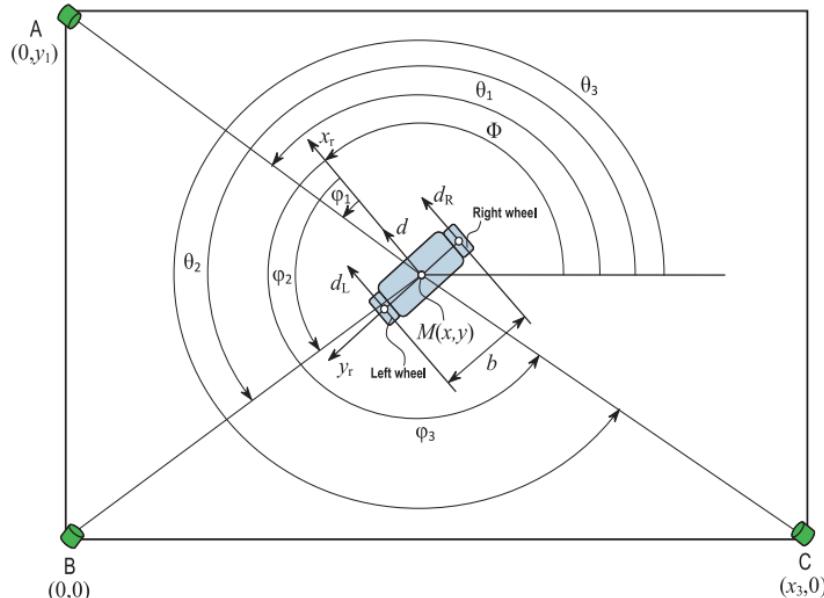


FIGURE 1 – Triangulation du robot

On décrit *l'état* du robot au pas de temps n par sa position dans le repère cartésien et son orientation. Ainsi, $\mathbf{x}_n = (x_n, y_n, \Phi_n)$. Pour mener à bien l'expérience, il nous faut construire :

- Le modèle de déplacement du robot
- Le modèle d'observation du robot par les beacons

2.1 Modèle de déplacement

Supposons que la distance incrémentale parcourue par la roue droite est d_R et gauche d_L . On détermine grâce à cela la distance parcourue à une itération et le changement d'orientation avec

$$d_n = \frac{d_{Ln} + d_{Rn}}{2} \quad (1)$$

$$\phi_n = \arctan\left(\frac{d_{Ln} - d_{Rn}}{b}\right) = \frac{d_{Ln} - d_{Rn}}{b} \quad (2)$$

Noter que (2) a pu être fait car l'argument de l'arctangente est supposé petit. Ainsi, à partir de l'ancienne position du robot et du nouvel ensemble de commandes, on a le modèle de déplacement suivant qui met à jour la variable d'état du robot :

$$x_n = x_{n-1} + d_n * \cos(\Phi_{n-1} + \phi_n) \quad (3)$$

$$y_n = y_{n-1} + d_n * \sin(\Phi_{n-1} + \phi_n) \quad (4)$$

$$\Phi_n = \Phi_{n-1} + \phi_n \quad (5)$$

Comme on peut le voir avec les équations (4) et (5), le modèle de déplacement est non linéaire en ses variables d'états. Cela posera problème plus tard dans le projet où il faudra linéariser pour pouvoir appliquer le filtre de Kalman.

2.2 Modèle d'observation

Les variables d'état sont observables via les mesures des 3 angles donnés par les 3 beacons φ_1, φ_2 et φ_3 .

Pour l'itération n, suivant le modèle 1, on

$$\varphi_{in} = \theta_{in} - \Phi_n$$

avec i pour chacun des 3 beacons.

Les θ sont calculés comme

$$\theta_{in} = \arctan\left(\frac{y_i - y_n}{x_i - x_n}\right)$$

et normalisés. On a donc $\mathbf{z}_n = (\varphi_1, \varphi_2, \varphi_3)$ qui est le vecteur obtenu à partir du modèle d'observation

Dès, lors les mesures "linéarisées" sont données en résolvant le système ci-dessus et on a

$$A_n = \frac{y_1 * \sin(\varphi_{3n} - \varphi_{2n})}{x_3 * \sin(\varphi_{2n} - \varphi_{1n})} \quad (6)$$

$$\tilde{\Phi}_n = \frac{A_n * \cos(\varphi_{1n}) - \sin(\varphi_{3n})}{A_n * \sin(\phi_{1n}) + \cos(\varphi_{3n})} \quad (7)$$

$$\tilde{x}_n = \frac{x_3 * \tan(\tilde{\Phi}_n + \varphi_{3n})}{\tan(\tilde{\Phi}_n + \varphi_{3n}) - \tan(\tilde{\Phi}_n + \varphi_{2n})} \quad (8)$$

$$\tilde{y}_n = \tilde{x}_n * \tan(\phi_{2n} + \tilde{\Phi}_n) \quad (9)$$

Dans le projet, on note $\mathbf{y}_n = (\tilde{x}_n, \tilde{y}_n, \tilde{\Phi}_n)$ comme le vecteur des positions estimées par la linéarisation du modèle d'observation.

3 Modèle d'état

On considère le modèle d'état suivant :

$$\mathbf{x}_n = \mathbf{f}_n(\mathbf{x}_{n-1}, \mathbf{u}_n, \mathbf{w}_n, \mathbf{e}_n) \quad (10)$$

$$\mathbf{z}_n = \mathbf{h}_n(\mathbf{x}_n, \mathbf{v}_n) \quad (11)$$

Dans ce modèle, $x_n \in \mathbb{R}^K$ est le vecteur d'état, $u_n \in \mathbb{R}^L$ est le vecteur d'entrée, $z_n \in \mathbb{R}^M$ est le vecteur de mesure, et $f_n(\cdot)$ et $h_n(\cdot)$ sont des fonctions non linéaires dépendant du temps. Nous supposons que toutes les composantes aléatoires sont des processus gaussiens blancs centrés et non corrélés. Plus précisément, le bruit de processus $w_n \in \mathbb{R}^P$, le bruit d'entrée $e_n \in \mathbb{R}^H$, et le bruit d'observation $v_n \in \mathbb{R}^M$ vérifient les propriétés suivantes :

$$\mathbb{E}\{w_n\} = 0, \quad \mathbb{E}\{e_n\} = 0, \quad \mathbb{E}\{v_n\} = 0,$$

$$\mathbb{E}\{w_i e_j^\top\} = 0, \quad \mathbb{E}\{w_i v_j^\top\} = 0, \quad \mathbb{E}\{v_j e_i^\top\} = 0 \quad \text{pour tout } i \text{ et } j.$$

Les matrices de covariance des bruits sont définies comme suit :

$$Q = \mathbb{E}\{w_n w_n^\top\}, \quad L = \mathbb{E}\{e_n e_n^\top\}, \quad R = \mathbb{E}\{v_n v_n^\top\}.$$

4 Modèle d'état étendu

Pour appliquer Kalman, il convient de linéariser le système.

Pour appliquer une technique telle que le filtrage de Kalman, les équations du modèle d'état doivent être développées en séries de Taylor d'ordre 1 ou 2. Nous utilisons uniquement les développements d'ordre 1 de $f_n(\cdot)$ au point $n-1$ et de $h_n(\cdot)$ au point n , sous les hypothèses suivantes.

Nous supposons que u_n varie suffisamment lentement, de sorte que la différence $u_n - u_{n-1}$ soit négligeable. De plus, nous considérons que les valeurs initiales sont connues, ce qui signifie que les composantes de bruit au point initial sont nulles.

Ainsi, les fonctions non linéaires développées deviennent :

$$\mathbf{x}_n = \mathbf{F}_n * \mathbf{x}_n + \bar{\mathbf{u}}_n + \mathbf{W}_n * \mathbf{w}_n + \mathbf{E}_n * \mathbf{E}_n \quad (12)$$

$$\mathbf{z}_n = \mathbf{H}_n * \mathbf{x}_n + \bar{\mathbf{z}}_n + \mathbf{v}_n \quad (13)$$

où $F_n = \frac{\partial f_n}{\partial x}\Big|_{x^{n-1}}$, $W_n = \frac{\partial f_n}{\partial W}\Big|_{x^{n-1}}$, $E_n = \frac{\partial f_n}{\partial e}\Big|_{x^{n-1}}$, $T_n = \frac{\partial h_n}{\partial v}\Big|_{x^{n-1}}$, et $H_n = \frac{\partial h_n}{\partial x}\Big|_{x^{n-1}}$ sont les matrices jacobiniennes, et les termes $\bar{u}_n = f_n(x^{n-1}, u_n, 0, 0) - F_n x^{n-1}$ et $\bar{z}_n = h_n(x^{n-1}) - H_n x^{n-1}$ sont connus. Ici, x^n est l'estimation à l'instant n et x^{n-1} est l'estimation antérieure de x_n . Dans l'article on assume que $\mathbf{F}_n = \mathbf{W}_n$ car le bruit de processus ω_n est additif aux composantes de \mathbf{x}_n dans le modèle de déplacement. En dérivant celui-ci, on a alors la matrice suivante :

$$\mathbf{F}_n = \begin{bmatrix} 1 & 0 & -d_n \sin\left(\hat{\Phi}_{n-1} + \frac{\phi_n}{2}\right) \\ 0 & 1 & d_n \cos\left(\hat{\Phi}_{n-1} + \frac{\phi_n}{2}\right) \\ 0 & 0 & 1 \end{bmatrix}$$

où les d_n , ϕ_n sont évalués grâce à (2).

On a aussi :

$$\mathbf{E}_n = \begin{bmatrix} E_{11n} & E_{12n} \\ E_{21n} & E_{22n} \\ -\frac{1}{b} & \frac{1}{b} \end{bmatrix}$$

où les éléments sont donnés par :

$$E_{11n} = \frac{1}{2} \cos \left(\hat{\Phi}_{n-1} + \frac{\phi_n}{2} \right) + \frac{d_n}{2b} \sin \left(\hat{\Phi}_{n-1} + \frac{\phi_n}{2} \right) \quad (14)$$

$$E_{12n} = \frac{1}{2} \cos \left(\hat{\Phi}_{n-1} + \frac{\phi_n}{2} \right) - \frac{d_n}{2b} \sin \left(\hat{\Phi}_{n-1} + \frac{\phi_n}{2} \right) \quad (15)$$

$$E_{21n} = \frac{1}{2} \sin \left(\hat{\Phi}_{n-1} + \frac{\phi_n}{2} \right) - \frac{d_n}{2b} \cos \left(\hat{\Phi}_{n-1} + \frac{\phi_n}{2} \right) \quad (16)$$

$$E_{22n} = \frac{1}{2} \sin \left(\hat{\Phi}_{n-1} + \frac{\phi_n}{2} \right) + \frac{d_n}{2b} \cos \left(\hat{\Phi}_{n-1} + \frac{\phi_n}{2} \right) \quad (17)$$

Enfin la matrice d'observation linéarisée est : Raisonnant de manière similaire, nous développons $h_n(x_n)$ à l'instant n sous la forme :

$$h_n(x_n) = H_n x_n + \bar{z}_n, \quad (18)$$

où $H_n = \frac{\partial h_n}{\partial x} \Big|_{x^{n-1}}$ est la matrice jacobienne, et

$$H_n = \begin{bmatrix} \frac{y_1 - \hat{y}_n^-}{(\hat{x}_n^-)^2 + (y_1 - \hat{y}_n^-)^2} & \frac{\hat{x}_n^-}{(\hat{x}_n^-)^2 + (y_1 - \hat{y}_n^-)^2} & -1 \\ \frac{-\hat{y}_n^-}{(\hat{x}_n^-)^2 + (\hat{y}_n^-)^2} & \frac{\hat{x}_n^-}{(\hat{x}_n^-)^2 + (\hat{y}_n^-)^2} & -1 \\ \frac{-\hat{y}_n^-}{(x_3 - \hat{x}_n^-)^2 + (\hat{y}_n^-)^2} & \frac{-x_3 + \hat{x}_n^-}{(x_3 - \hat{x}_n^-)^2 + (\hat{y}_n^-)^2} & -1 \end{bmatrix}$$

et $\bar{z}_n = h_n(\hat{x}_{n-1}) - H_n \hat{x}_{n-1}$ est connu.

Avec ce nouveau modèle d'état étendu nous sommes capables d'établir l'EKF et l'EFIR.

5 EKF

Une première remarque que l'on peut émettre à propos de l'article est qu'il ne spécifie pas l'algorithme de l'EKF utilisé, contrairement à [2]. En fait, il ne fait pratiquement aucune analyse du filtre EKF, de ses inconvénients ou avantages, à part lors de la comparaison avec EFIR. Nous avons repris l'algorithme fourni par cette source annexe, qui est celui-ci dessous.

Input: $\mathbf{z}_n, \hat{\mathbf{x}}_0, \mathbf{P}_0, \mathbf{R}, \mathbf{Q}, \mathbf{L}$

- 1: **for** $n = 1 : M$ **do**
- 2: $\hat{\mathbf{x}}_n^- = \mathbf{f}_n(\hat{\mathbf{x}}_{n-1}, \mathbf{u}_n, \mathbf{0}, \mathbf{0})$
- 3: $\mathbf{P}_n^- = \mathbf{F}_n \mathbf{P}_{n-1} \mathbf{F}_n^T + \mathbf{W}_n \mathbf{Q} \mathbf{W}_n^T + \mathbf{E}_n \mathbf{L} \mathbf{E}_n^T$
- 4: $\mathbf{K}_n = \mathbf{P}_n^- \mathbf{H}_n^T (\mathbf{H}_n \mathbf{P}_n^- \mathbf{H}_n^T + \mathbf{T}_n \mathbf{R}_n \mathbf{T}_n^T)^{-1}$
- 5: $\hat{\mathbf{x}}_n = \hat{\mathbf{x}}_n^- + \mathbf{K}_n [\mathbf{z}_n - \mathbf{h}_n(\hat{\mathbf{x}}_n^-, \mathbf{0})]$
- 6: $\mathbf{P}_n = (\mathbf{I} - \mathbf{K}_n \mathbf{H}_n) \mathbf{P}_n^-$
- 7: **and for**

Output: $\hat{\mathbf{x}}_n$

FIGURE 2 – Algorithme d'EKF utilisé

L'inconvénient de l'EKF est qu'il faut fournir de l'information quant aux matrices R,Q,L. On note que M dans la figure 2 est pour le nombre d'itérations. Pour déterminer les matrices R,Q,L nous avons fait comme dans l'article c'est à dire :

$$Q = \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_\Phi^2 \end{bmatrix} = \begin{bmatrix} 10^{-4} & 0 & 0 \\ 0 & 10^{-4} & 0 \\ 0 & 0 & 7.62 \times 10^{-5} \end{bmatrix}$$

$$L = \begin{bmatrix} \sigma_L^2 & 0 \\ 0 & \sigma_R^2 \end{bmatrix} = \begin{bmatrix} 10^{-4} & 0 \\ 0 & 10^{-4} \end{bmatrix}$$

$$R = \begin{bmatrix} \sigma_\phi^2 & 0 & 0 \\ 0 & \sigma_\phi^2 & 0 \\ 0 & 0 & \sigma_\phi^2 \end{bmatrix} = \begin{bmatrix} 1.218 \times 10^{-3} & 0 & 0 \\ 0 & 1.218 \times 10^{-3} & 0 \\ 0 & 0 & 1.218 \times 10^{-3} \end{bmatrix}$$

On utilise comme dans l'application de la référence un facteur de correction p égal à 5 dont le carré multiplie Q et divise L et Q. L'algorithme a besoin du vecteur d'observation \mathbf{z}_n qui est connu car il contient les mesures par les beacons. Pour l'exemple ci-dessous, le robot démarre avec les conditions initiales $\mathbf{x}_0 = (0, 0, 0)$ et la matrice de covariance initiale est l'identité de dimension 3.

Pour une première simulation de l'EKF, donnons comme estimation initiale la position réelle du robot ainsi que la bonne matrice de covariance.

FIGURE 3 – Algorithme d'EKF pour prédire la position du robot avec de bonnes conditions initiales

On voit que l'estimation par cette méthode dans ces conditions est excellente. On peut maintenant essayer avec des mauvaises conditions initiale et voir ce qui se passe. Mettons que nous prenons l'estimation de la position initiale à (10,20,0) au lieu de (0,0,0).

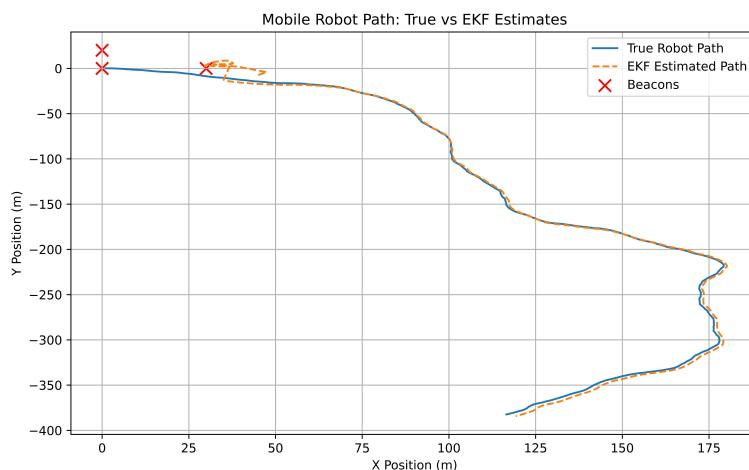


FIGURE 4 – Algorithme d'EKF pour prédire la position du robot avec de mauvaises conditions initiales

Cependant, noter que parfois l'estimation peut diverger :moins d'une exécution sur 10 si les conditions initiales sont bonnes et près d'une fois sur 4 pour le cas de la mauvaise condition initiale que nous avons testée. Cela s'explique par plusieurs facteurs :

1. **Conditions initiales incorrectes** : Une mauvaise estimation des états ou des covariances d'erreur peut entraîner la divergence ou la mauvaise convergence du filtre.
2. **Non-linéarités excessives** : L'EKF linéarise les modèles non linéaires, mais si la non-linéarité est trop forte, l'approximation devient inefficace, ce qui perturbe les estimations.
3. **Bruit de mesure ou de processus élevé** : Si les niveaux de bruit sont mal estimés ou trop importants, l'EKF peut donner des résultats erronés, surtout dans des conditions bruyantes.
4. **Mauvaise modélisation** : Si le modèle de transition ou d'observation n'est pas bien conçu ou trop simplifié, l'EKF peut échouer à suivre correctement l'état du système.

6 EFIR

L'algorithme EFIR (Extended Finite Impulse Response) est une approche de filtrage non linéaire qui est une alternative aux filtres de Kalman traditionnels, tels que l'Extended Kalman Filter (EKF). Contrairement aux méthodes classiques, l'EFIR ne repose pas sur des hypothèses de linéarité et ignore les statistiques du bruit de mesure et de processus, ce qui le rend robuste dans des environnements de bruit non Gaussien, notamment lorsque les bruits sont de type "heavy-tailed" ou avec des valeurs aberrantes.

Pour cet algorithme, nous reprenons celui qui est donné dans [1].

```

Input:  $\mathbf{z}_n, \mathbf{y}_n, K, N$ 
1:   for  $n = N - 1 : M$    do
2:      $m = n - N + 1, s = m + K - 1$ 
3:      $\tilde{\mathbf{x}}_s = \begin{cases} \mathbf{y}_s, & \text{if } s < N - 1 \\ \hat{\mathbf{x}}_s, & \text{if } s \geq N - 1 \end{cases}$ 
4:      $\mathbf{G}_s = \mathbf{F}_s \mathbf{F}_{s-1} (\mathbf{H}_{s,m}^T \mathbf{H}_{s,m})^{-1} \mathbf{F}_{s-1}^T \mathbf{F}_s^T$ 
5:     for  $l = m + K : n$    do
6:        $\tilde{\mathbf{x}}_l^- = \mathbf{f}_l(\tilde{\mathbf{x}}_{l-1}, \mathbf{u}_l, \mathbf{0}, \mathbf{0})$ 
7:        $\mathbf{G}_l = [\mathbf{H}_l^T \mathbf{H}_l + (\mathbf{F}_l \mathbf{G}_{l-1} \mathbf{F}_l^T)^{-1}]^{-1}$ 
8:        $\mathbf{K}_l = \mathbf{G}_l \mathbf{H}_l^T$ 
9:        $\tilde{\mathbf{x}}_l = \tilde{\mathbf{x}}_l^- + \mathbf{K}_l [\mathbf{z}_l - \mathbf{h}_l(\tilde{\mathbf{x}}_l^-)]$ 
10:      and for
11:         $\hat{\mathbf{x}}_n = \tilde{\mathbf{x}}_n$ 
12:      and for
Output:  $\hat{\mathbf{x}}_n$ 

```

FIGURE 5 – Algorithme d'EFIR

On peut expliquer cet algorithme dans les grandes lignes.

L'algorithme EFIR (Efficient Finite Impulse Response) fonctionne en analysant les mesures z_n dans un intervalle glissant de longueur N , allant de $m = n - N + 1$ à n . À chaque étape, il combine un modèle de mouvement et les observations pour fournir une estimation robuste de l'état \hat{x}_n .

— **Initialisation de l'état \tilde{x}_s :**

- Si $s \leq N - 1$, les estimations sont directement obtenues à partir des états observés y_s .
- Si $s > N - 1$, l'estimation précédente \tilde{x}_s est utilisée.

— **Lissage initial (G_s) :**

- Pour éviter les singularités, le gain G_s est calculé en mode batch, en combinant les modèles de mouvement (F_s) et d'observation (H_s, m).

— **Prédiction :**

- L'état prédit \tilde{x}_l^- est calculé à l'aide du modèle de mouvement f_l .

— **Mise à jour des gains (G_l, K_l) :**

- G_l est le gain calculé en ignorant les covariances de bruit, ce qui simplifie l'algorithme.
- K_l , le gain de correction, ajuste les prédictions \tilde{x}_l^- en fonction des observations.

— **Mise à jour de l'état :**

- L'estimation corrigée \tilde{x}_l intègre les résidus de mesure via : $[\tilde{x}_l = \tilde{x}_l^- + K_l [z_l - h_l(\tilde{x}_l^-)]]$

— **État final :**

- Après toutes les mises à jour, \hat{x}_n est produit comme l'état estimé final.

Une deuxième remarque par rapport à l'article ici est qu'il ne donne pas la forme de \mathbf{H}_{sm} . Pour cela, on s'aide de [2].

$$\begin{bmatrix} \mathbf{H}_s \mathbf{F}_s \cdots \mathbf{F}_{m+1} \\ \vdots \\ \mathbf{H}_{m+1} \mathbf{F}_{m+1} \\ \mathbf{H}_m \end{bmatrix}.$$

Dans le cadre de l'algorithme EFIR (Extended Finite Impulse Response), la matrice

$$\mathbf{G}_s$$

peut être remplacée par une matrice identité \mathbf{I} dans de nombreux cas. En effet, le GNPG (Gain Normalized Propagation Gain), qui mesure l'effet de propagation des gains, est presque égal à 1 sur un intervalle de K points. Cette simplification facilite les calculs tout en restant précise, car \mathbf{G}_s n'ajoute pas d'effet significatif lorsque le GNPG est proche de l'unité. Lors de notre application, cela a permis de stabiliser les résultats.

Un autre manquement à l'article est le fait qu'il ne mentionne pas la "dead zone". En effet, il est impossible de calculer l'efir pour les $N-1$ premiers points. Nous avons dès lors décidé d'utiliser l'EKF pour le calcul de ces premiers points, comme cela a été fait dans [2].

Une autre chose qu'il faut faire pour l'efir est de déterminer le nombre N_{opt} . Pour cela, on minimise la trace de la matrice $\mathbf{P}(n)$ qui est calculée comme

$$\mathbf{P}_n = \mathbb{E} \{ (\mathbf{x}_n - \tilde{\mathbf{x}}_n) * (\mathbf{x}_n - \tilde{\mathbf{x}}_n) \}$$

On fait alors tourner l'algorithme pour plusieurs valeurs de N et on garde la valeur de N qui minimise la trace pour l'application.

Pour une première simulation, on reprend les mêmes conditions que l'article et on compare efir et ekf.

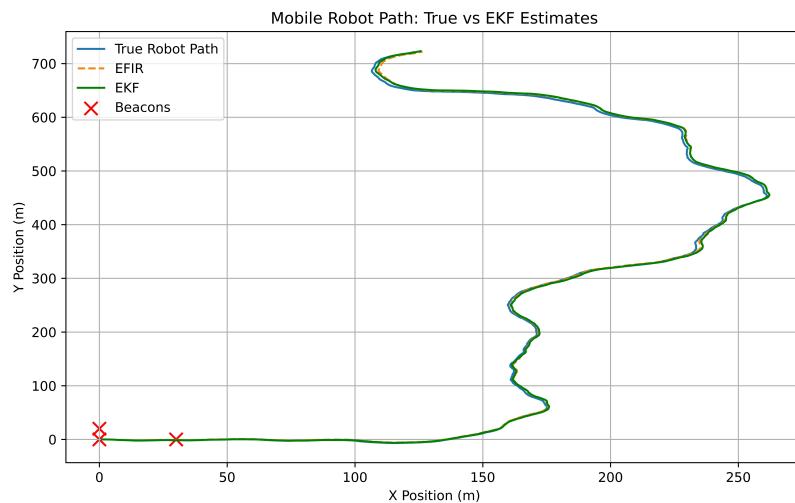


FIGURE 6 – Algorithme d’EFIR vs EKF, seed 24

On peut voir ici que les 2 offrent des résultats comparables comme dans l’article.

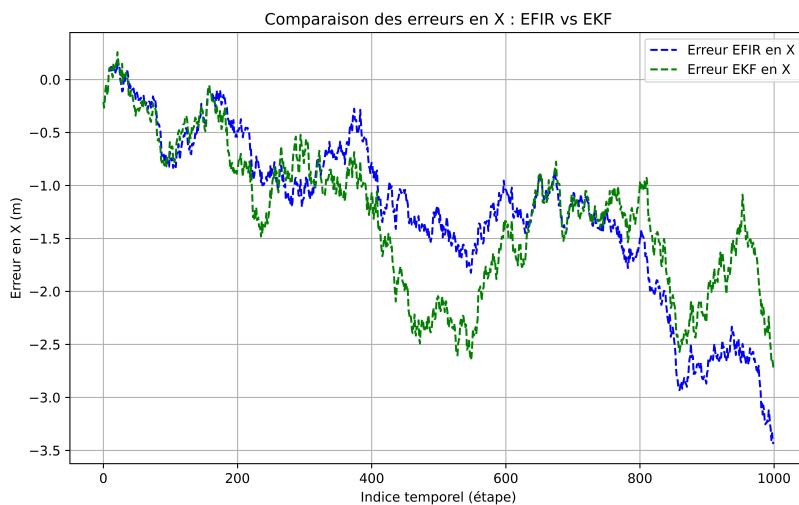


FIGURE 7 – Comparaison des coordonnées x

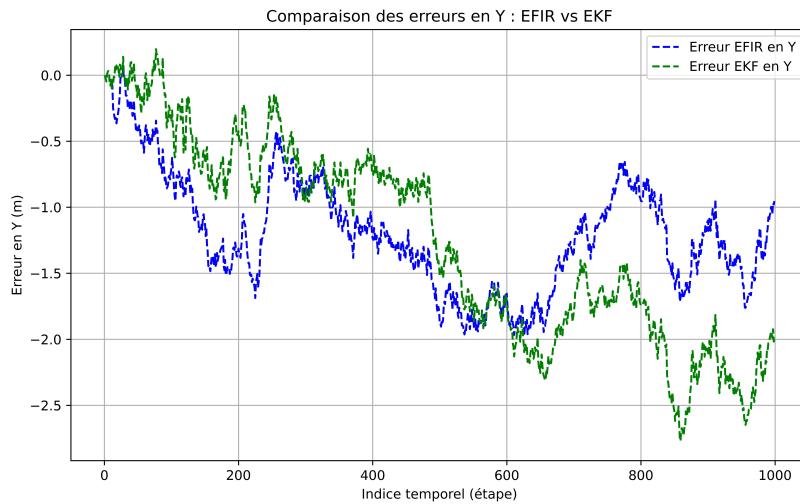


FIGURE 8 – Comparaison des coordonnées y

Exploitons maintenant les différences entre les deux algorithmes.

On va tout d'abord changer le bruit. On rappelle que seul l'ekf est influencé par les statistiques de bruit. Prenons les 3 matrices de bruit Q,R et L et on multiplie tous leurs éléments par 2. On a alors le graphique suivant

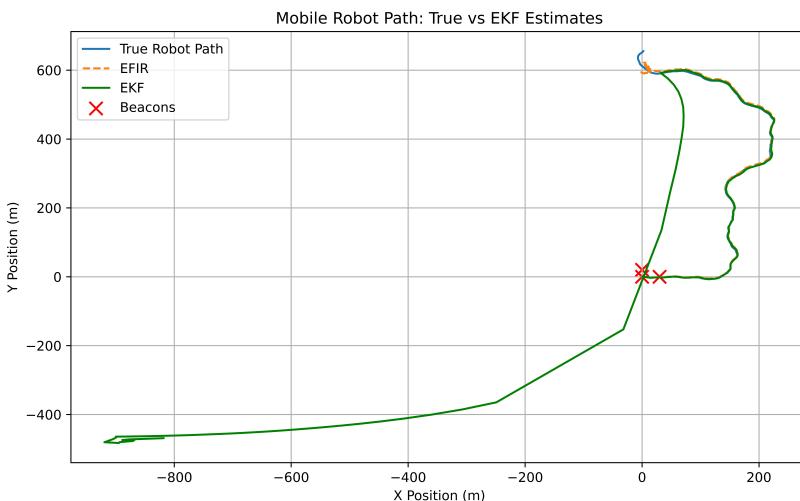


FIGURE 9 – Algorithme d'EFIR vs EKF, seed 22

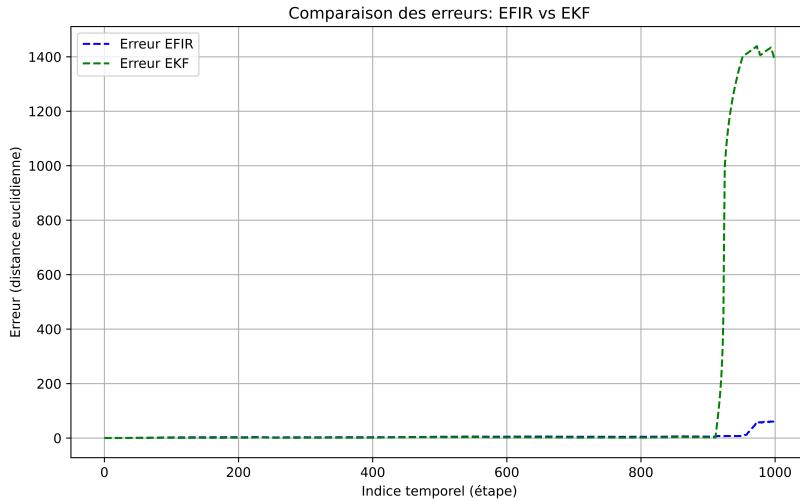


FIGURE 10 – Erreur d’EFIR vs EKF pour des données plus bruitées, seed 22

On a pu constater au cours de nos nombreuses simulations que l’efir est moins sensible au bruit. Cependant, il reste légèrement dépendant car on rappelle que nous avons utilisé ekf pour calculer les premiers points de l’efir.

Enfin, on change les conditions initiales pour voir la convergence des algorithmes.

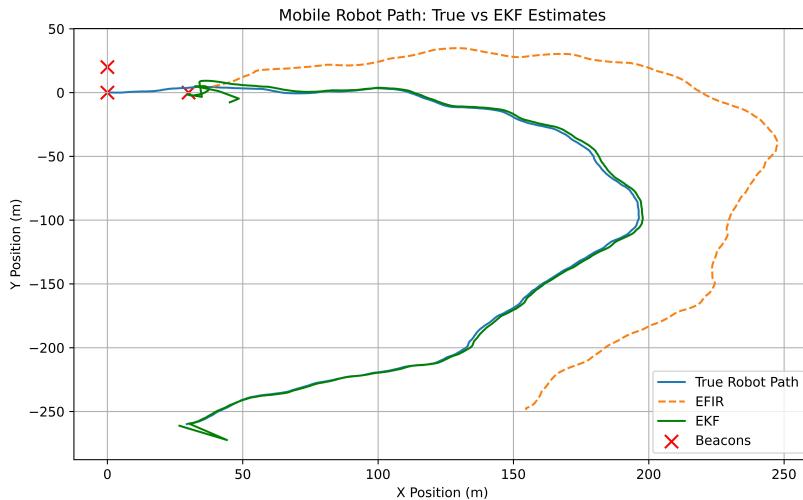


FIGURE 11 – Algorithme d’EFIR vs EKF, seed 22

On voit qu’ici, c’est l’ekf qui s’en sort mieux si les conditions initiales sont mal estimées. Cela n’a pas été illustré dans l’article.

7 Conclusion

Ce projet a permis d’explorer deux méthodes de filtrage pour la localisation d’un robot mobile, à savoir le filtre de Kalman étendu (EKF) et le filtre EFIR. En partant d’un modèle de déplacement et d’observation non linéaire, nous avons mis en œuvre les étapes de linéarisation

nécessaires pour appliquer l'EKF. L'analyse a révélé que l'EKF offre de bonnes performances lorsque les conditions initiales et les modèles statistiques du bruit sont correctement définis. Cependant, ses limites, notamment sa sensibilité au bruit et aux erreurs d'estimation initiales, le rendent moins robuste dans des environnements incertains.

En comparaison, le filtre EFIR, qui ne dépend pas des hypothèses statistiques du bruit, pourrait présenter des avantages en termes de robustesse. Néanmoins, ses performances doivent être examinées plus en détail pour évaluer sa précision et son efficacité dans des conditions variées.

Ainsi, cette étude souligne l'importance du choix du filtre en fonction des spécificités de l'application. Elle ouvre également la voie à des investigations futures sur l'intégration de ces méthodes dans des systèmes robotiques réels ou sur le développement de nouvelles approches hybrides combinant les atouts des deux filtres.

Références

- [1] Triangulation-Based Indoor Robot Localization Using Extended FIR/Kalman Filtering, Moises Granados-Cruz, Juan Pomarico-Franquiz, Yuriy S. Shmaliy, Luis J. Morales-Mendoza, Universidad de Guanajuato, Universidad Veracruzana
- [2] S. Zhao, J. Pomárico-Franquiz and Y. S. Shmaliy, "An approach to nonlinear state estimation using extended FIR filtering," 2014 22nd European Signal Processing Conference (EUSIPCO), Lisbon, Portugal, 2014, pp. 436-440. keywords : Noise ;Hidden Markov models ;Kalman filters ;Noise measurement ;Estimation error ;Vectors ;State-space methods
- [3] Combined extended FIR/Kalman filtering for indoor robot localization via triangulation, Juan Pomárico-Franquiz a, Sanowar H. Khan b, Yuriy S. Shmaliy a <https://www.sciencedirect.com/science/article/abs/pii/S0263224114000062>
- [4] Robust and accurate UWB-based indoor robot localisation using integrated EKF/EFIR filtering, Yuan Xu, Yuriy S. Shmaliy, Choon Ki Ahn, Guohui Tian, Xiyuan Chen <https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/iet-rsn.2017.0461>