ROB312 - Navigation pour les systèmes autonomes
TP2 Filtrage de Kalman pour la navigation

SANTOS SANO Matheus

Palaiseau, France

Octobre 2023

Exercice 1

Le code est une implémentation du filtre de Kalman étendu (EKF) pour la localisation d'un robot. Pour ce faire, on considère un système réel composé d'un robot avec un odomètre qui calcule la position du robot. Pour prédire la position du robot, le filtre de Kalman est implementé et il se compose de deux parties, la prédiction et la correction.

Dans la phase de prédiction, on implemente un modèle mathématique pour estimer le prochain état $\hat{x}_{k|k-1}$ à partir de l'état estimé précédent $\hat{x}_{k-1}$, de la position bruyante calculée par l'odomètre u_k et le temps entre les deux prévisions Δt :

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1}, u_k) \tag{1}$$

Comme le modèle mathématique n'est pas linéaire, l'état estimé ne suit pas une distribution gaussienne. Par conséquent, l'algorithme du filtre de Kalman ne converge pas. Il faut donc implémenter un filtre de Kalman étendu (EKF), qui linéarise la fonction non linéaire autour de la moyenne de l'état actuel estimé. À chaque pas de temps k, la linéarisation est effectuée localement et les matrices jacobiennes résultantes sont utilisées dans les états de prédiction et de correction de l'algorithme du filtre de Kalman.

Toujours au stade de la prédiction, on calcule la prédiction de la covariance $\hat{P}_{k|k-1}$, qui représente l'incertitude associée à la prédiction de l'état. Elle est obtenue à partir de la matrice de covariance de l'erreur de processus Q_k et des matrices jacobiennes de la fonction de prédiction en fonction de l'état F_k et du bruit G_k :

$$\hat{P}_{k|k-1} = F_k \hat{P}_{k-1} F_k^T + G_k Q_k G_k^T \tag{2}$$

Dans la section de correction, l'état du système (robot) z_k est mesuré à partir de l'odomètre. Il faut noter que l'état calculé par l'odomètre est bruité, et ce bruit est représenté par la matrice de covariance de l'erreur de mesure R_k.

Le filtre de Kalman cherche à éliminer l'erreur entre la mesure et l'estimation via le gain de Kalman K_k. Le calcul du gain de Kalman dépend de la prédiction de la covariance $P_{k|k-1}$, de la matrice jacobienne de la fonction d'observation H_k et de la covariance de l'erreur de mesure R_k :

$$K_k = \hat{P}_{k|k-1} H_k^T S_k^{-1}, \qquad S_k = R_k + H_k \hat{P}_{k|k-1} H_k^T \tag{3}$$

Ainsi, sur la base du gain de Kalman K_k, de l'état mesuré par l'odomètre z_k et de la fonction d'observation $h(\hat{x}_{k|k-1})$, l'état estimé $\hat{x}_k$ est mis à jour afin de minimiser l'erreur entre la mesure et l'estimation :

$$\hat{x}_k = \hat{x}_{k|k-1} + K_k(z_k - h(\hat{x}_{k|k-1})) \tag{4}$$

En outre, la covariance de l'erreur d'estimation P_k est mise à jour sur la base du gain de Kalman afin de réduire l'incertitude après la correction de l'état estimé :

$$\hat{P}_k = (I_d - K_k H_k)\hat{P}_{k|k-1} \tag{5}$$

Comme le filtre de Kalman est un processus récursif, l'algorithme revient à l'étape de prédiction après avoir corrigé et mesuré à nouveau l'odomètre.

La première partie du code est l'implémentation de la classe appelée **Simulation** qui simule le mouvement du robot. Dans cette classe, les paramètres de simulation sont initialisés,

tels que le temps de simulation (**Tf**), l'intervalle de temps entre deux prédictions (**dt_pred**), l'intervalle de temps entre deux mesures (**dt_meas**), la position réelle du robot (**xTrue**), la matrice de covariance de l'erreur de processus et de l'erreur de mesure (**QTrue** et **RTrue**), la position odométrique (**xOdom**) et la carte des amers (**Map**). En outre, cette classe possède des méthodes pour obtenir le contrôle réel du robot à l'instant k (*get_robot_control*), simuler le monde avec la position réelle du robot à l'instant k (*simulate_world*), obtenir les mesures d'odométrie bruitées u_k (*get_odometry*) et mesurer l'état bruité du système z_k (*get_observation*).

L'autre partie du code consiste à implémenter les fonctions du filtre de Kalman. La fonction *motion_modèle* calcule la prédiction de l'état suivant du robot $hatx_{k|k-1}$ en utilisant la fonction $f(\cdot)$ (1). La fonction *observe_model* calcule l'état du système z_k à partir de l'état du robot **xVeh** et d'un point de référence. Le point de référence est choisi au hasard à chaque étape k de la matrice **Map**. L'indice du point de référence actuel est **iFeature**.

Comme le modèle mathématique n'est pas linéaire, la fonction *get_obs_jac* calcule la matrice jacobienne de la fonction d'observation H_k. En outre, les fonctions F et G calculent, respectivement, les matrices jacobiennes de la fonction de prédiction en fonction de l'état F_k et du bruit G_k.

En outre, certaines fonctions sont implémentées, telles que *plot_covariance_ellipse*, qui trace une ellipse d'erreur à partir de la matrice de covariance, *angle_wrap*, qui ajuste l'angle entre $-\pi$ et π, et *tcomp*, qui effectue deux transformations représentant le modèle mathématique $f(\cdot)$.

La dernière partie du code est constituée du programme principal (main). Dans cette partie, divers paramètres sont initialisés, tels que les temps de simulation, l'estimation initiale de l'état du système $\hat{x}_0$, l'incertitude associée à l'estimation initiale de l'état $\hat{P}_0$, la carte des points de référence **Map** et les matrices de covariance **QEst** et **REst**. L'état est représenté par la position sur l'axe x, l'axe y et l'orientation θ.

Le programme principal lance le cycle de simulation, dans lequel, à chaque itération de la boucle, les étapes de prédiction et de correction sont exécutées. Dans la partie prédiction, l'algorithme simule le mouvement du robot (*simulate_world*), obtient les mesures d'odométrie (*get_odometry*), calcule les jacobiens F_k et G_k (F et G), prédit la position du robot $\hat{x}_{k|k-1}$ (**??**) (*motion_model*) et calcule la prédiction de la covariance $\hat{P}_{k|k-1}$ (2).

Dans la partie correction du filtre de Kalman, l'état du système z_k est obtenu à partir du point de référence dont l'index est **iFeature** (*get_observation*) et la fonction d'observation $h(\hat{x}_{k|k-1})$ est calculée (*observation_model*). Ensuite, la matrice jacobienne d'observation H_k est obtenue (*get_obs_jac*) et, à partir de H_k, le gain de Kalman K_k (3) est calculé.

Ainsi, l'état estimé $\hat{x}_k$ (4) et la covariance de l'erreur d'estimation P_k (5) sont mis à jour. En outre, le code conserve un historique des poses estimées et réelles du robot, des erreurs et des covariances afin de fournir une visualisation de la trajectoire du robot, de l'odométrie, de la pose estimée et des graphiques d'erreur.

En résumé, ce code est une simulation complète de localisation basée sur l'EKF pour un robot mobile opérant dans un environnement 2D avec des amers. Il fournit un retour visuel sur la précision de l'estimation de la pose du robot par l'EKF au fil du temps. Lorsque on exécute le code, il génère des graphiques montrant la trajectoire du robot, la trajectoire estimée et la covariance de l'erreur.

Exercice 2

Comme le montre la Figure 1, le code trace l'environnement de simulation 2D avec les points de référence (points noirs), la trajectoire réelle du robot (True), la trajectoire estimée du robot basée sur les prédictions du modèle odométrique (odom) et la trajectoire estimée du robot basée sur le filtre de Kalman étendu (EKF). En outre, le code trace une ellipse de covariance autour de l'état actuel du robot pour indiquer l'incertitude de l'estimation (ligne rouge en pointillé).

En outre, trois graphiques sont tracés pour montrer les erreurs dans l'estimation de l'état en fonction du temps pour les coordonnées x, y et l'angle θ. La courbe bleue montre l'erreur d'estimation de l'état du robot par rapport à la position réelle du robot. Ainsi, lorsque l'erreur est faible, l'estimation est proche de la position réelle du robot et, par conséquent, le filtre de Kalman estime correctement la position réelle du robot. Les courbes rouges, à son tour, représentent les limites de ± 3 fois l'écart-type de l'estimation de l'état et montrent à quel point l'estimation peut être dispersée par rapport à la position réelle du robot.

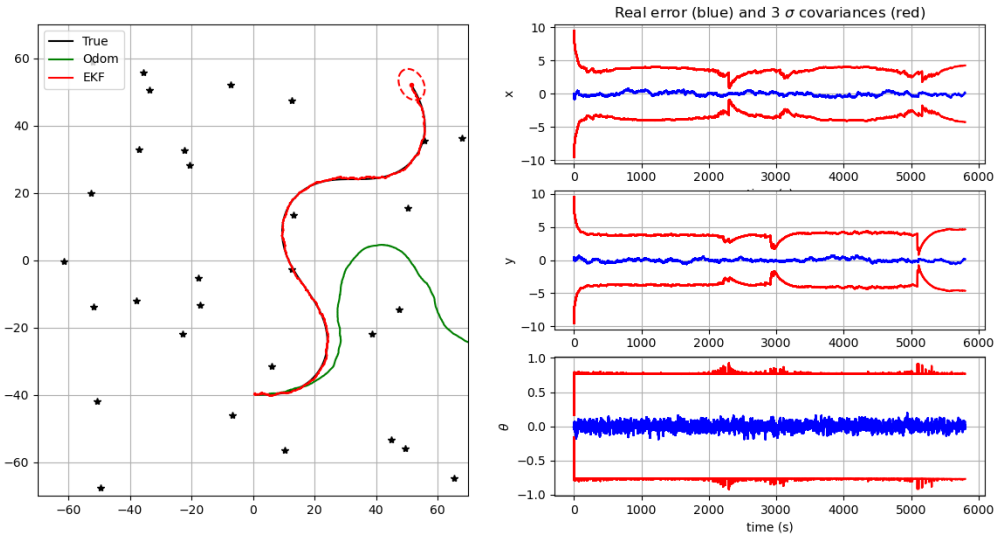

FIGURE 1 – Trajectoire estimée par l'odométrie (odom) et par le filtre de Kalman étendu (EKF). Erreurs dans l'estimation de l'état en fonction du temps pour les coordonnées x, y et l'angle θ.

Comme le montre la Figure 1, la trajectoire estimée du robot basée sur le filtre de Kalman (EKF) parvient à suivre correctement la trajectoire réelle du robot (True), ce qui témoigne de la bonne fonctionnalité du filtre de Kalman implémenté. Ceci est également visible dans les graphiques d'erreur, où l'erreur dans l'estimation de l'état du robot par rapport à la position réelle du robot (courbe bleue) est faible.

Il convient de noter que lorsque le robot passe à proximité d'un point de référence, le filtre de Kalman mesure l'état du robot et corrige donc les valeurs de position. Cet effet est visible dans les graphiques d'erreur, où l'on observe une forte diminution de la covariance lorsque le robot est proche d'un point de référence.

En outre, au fil du temps, la trajectoire estimée par l'odométrie (odom) commence à s'écarter de la trajectoire réelle du robot (True), ce qui indique que le modèle odométrique ne suit plus avec précision le mouvement réel du robot et que le filtre de Kalman doit corriger pour que sa prédiction (EKF) suive correctement la trajectoire réelle.

Exercice 3

La Figure 1 montre le fonctionnement du filtre avec $dt\_meas = 1$. Lorsque on augmente dt_meas, la fréquence des mesures diminue. Par conséquent, le filtre de Kalman reçoit moins d'informations pour corriger l'estimation de l'état entre deux mesures consécutives. Par conséquent, l'incertitude (covariance) de l'estimation de l'état augmente, car le filtre dispose de moins d'informations pour réduire cette incertitude. Ceci peut être vérifié en augmentant $dt\_meas = 10$ (Figure 2).

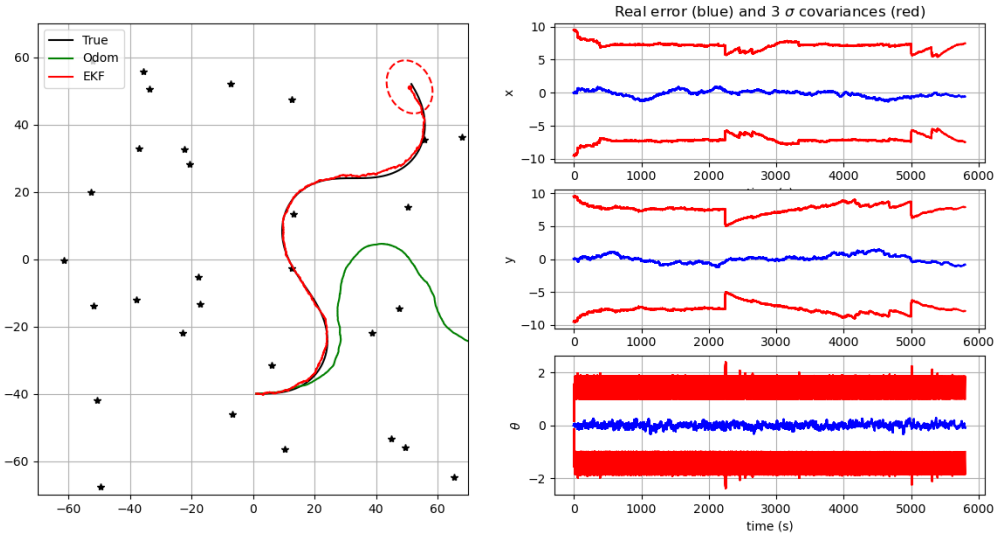

FIGURE 2 – Plot du code avec $dt\_meas = 10$.

En comparant la Figure 1 avec la Figure 2, on peut noter qu'en augmentant le intervalle de temps entre deux mesures dt_meas, la covariance et l'erreur dans l'estimation effectivement augmentent. Cette augmentation de l'erreur est visible dans la trajectoire estimée par le filtre de Kalman étendu (EKF), qui ne suit plus la trajectoire réelle du robot avec la même fidélité, surtout si on la compare au cas où $dt\_meas = 1$.

Cette erreur est d'autant plus importante que l'on augmente la valeur de dt_meas. Lorsque, par exemple, $dt\_meas = 100$ (Figure 3), la trajectoire estimée par l'EKF s'éloigne encore plus de la trajectoire réelle, ce qui entraîne une erreur d'estimation considérablement plus élevée. En outre, on remarque que l'ellipse de covariance s'élargit, ce qui reflète la plus grande incertitude de l'estimation.

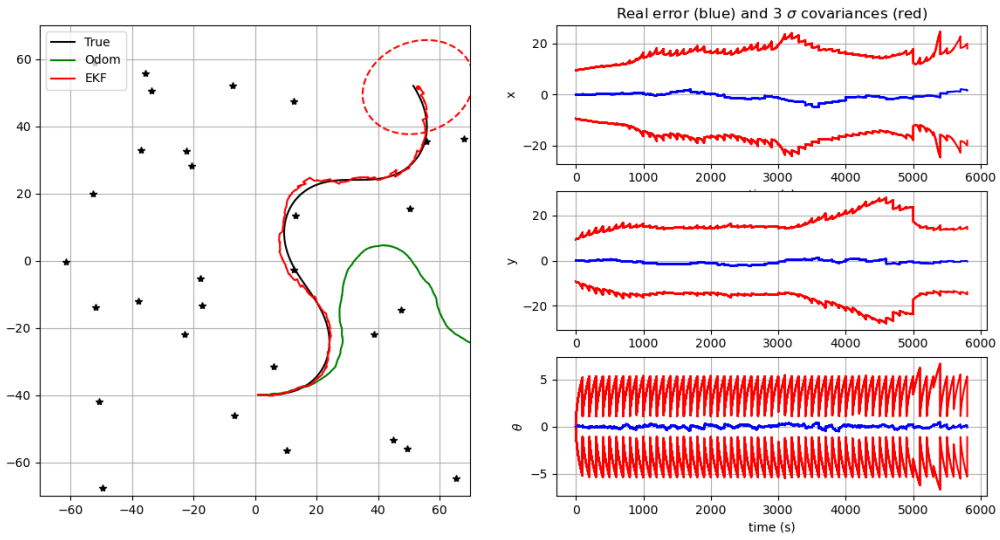

FIGURE 3 – Plot du code avec $dt\_meas = 100$.

Exercice 4

Comme la matrice QEst représente la covariance de l'erreur de processus, c'est-à-dire le bruit dynamique du filtre, sa variation affectera l'estimation du filtre de Kalman. En diminuant les valeurs présentes dans QEst, le modèle d'odométrie est considéré comme plus précis et, par conséquent, le processus a moins de bruit. De cette façon, l'estimation a tendance à suivre de plus près les prédictions du modèle et, par conséquent, l'incertitude de l'estimation est plus faible. Ceci peut être analysé dans la Figure 4.

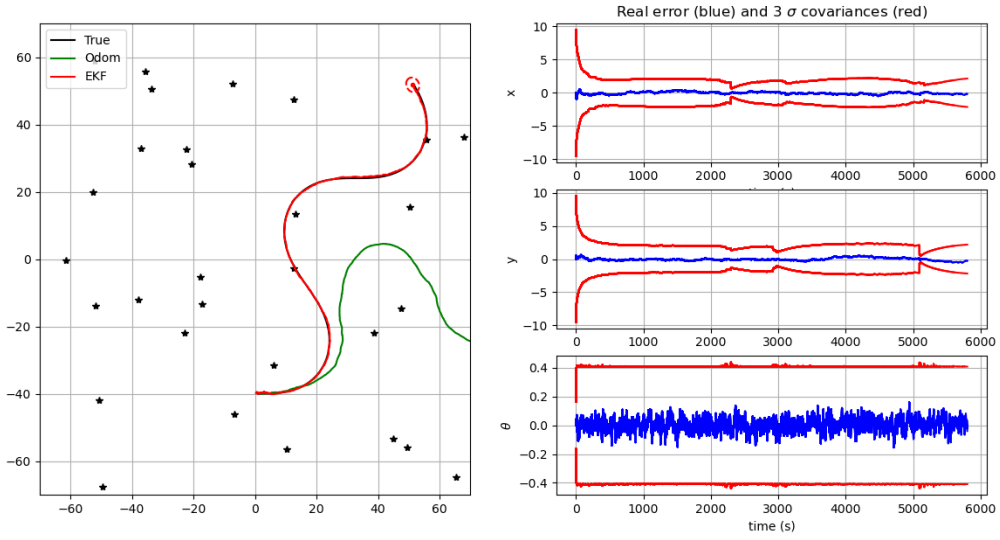

FIGURE 4 – Plot du code avec QEst 10 fois plus petit que celui indiqué dans la Figure 1

En comparant avec la Figure 1, on peut constater que l'erreur d'estimation et l'incertitude diminuent lorsque le bruit du processus est 10 fois inférieur (Figure 4). Cela se voit également dans l'illustration de gauche, où la trajectoire estimée par le filtre de Kalman (EKF) est plus précise par rapport la trajectoire réelle du robot, et l'ellipse d'incertitude est beaucoup plus petite, reflétant une plus grande précision dans le modèle d'odométrie. Cependant, si l'on réduit

trop les valeurs de QEst, le filtre attribue une faible incertitude au modèle d'odométrie et s'appuie fortement sur les prédictions générées par le modèle d'odométrie pour estimer l'état du robot. De cette façon, le filtre de Kalman sera plus sensible aux erreurs du modèle d'odométrie, comme le montre la Figure 5.

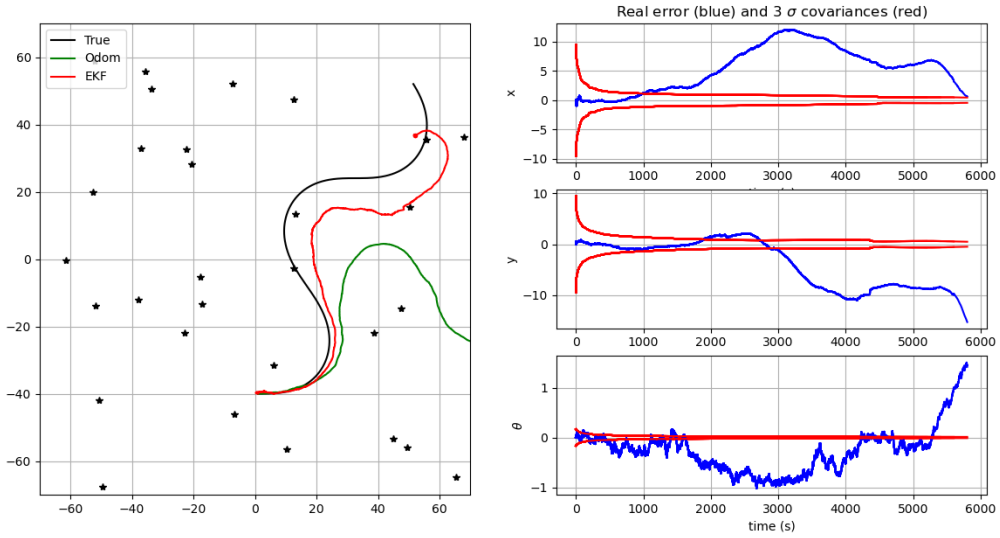

FIGURE 5 – Plot du code avec QEst 10^7 fois plus petit que celui indiqué dans la Figure 1

Comme le confirme la Figure 5, en réduisant considérablement le bruit dynamique, le filtre commence à faire beaucoup confiance aux prédictions générées par le odométrie et, ainsi, la trajectoire prédite par le filtre (EKF) commence à se rapprocher de celle prédite par le odométrie (odom). De cette façon, le filtre de Kalman devient plus sensible aux erreurs, ce qui augmente considérablement l'erreur d'estimation de l'état du robot par rapport à la position réelle.

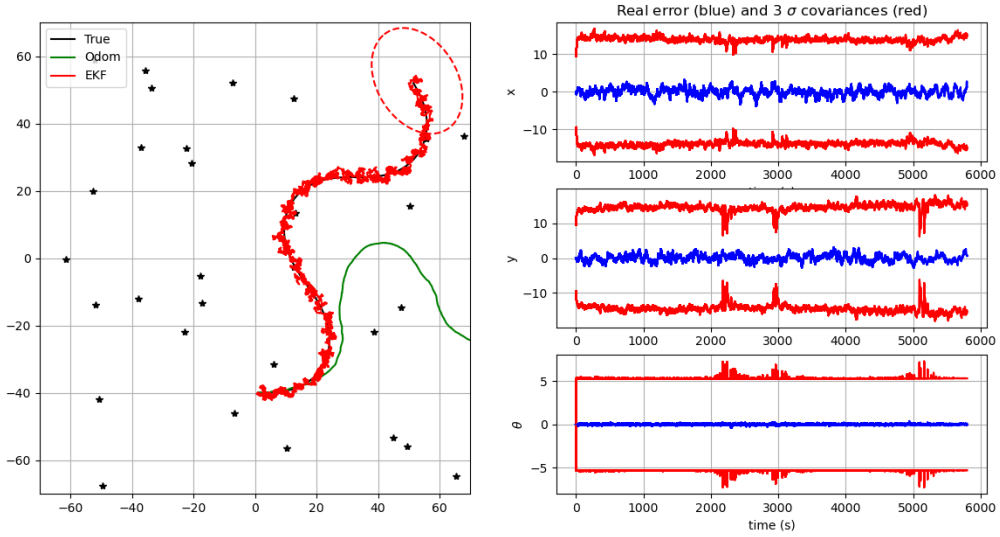

FIGURE 6 – Plot du code avec QEst 100 fois plus grand que celui indiqué dans la Figure 1

En revanche, en augmentant le bruit dynamique du filtre QEst, le modèle d'odométrie est considéré comme moins précis par le filtre et, ainsi, les mesures ont une influence plus significative sur l'estimation. De cette façon, toute mesure inexacte ou bruyante aura un impact direct sur l'estimation, conduisant à une plus grande incertitude, comme le montre la Figure 6.

Exercice 5

Alors que QEst représente l'incertitude associée au modèle d'odométrie, REst représente l'erreur des mesures du filtre. Par conséquent, si REst est très grand, le filtre attribue de grandes incertitudes aux mesures, ce qui donne plus de confiance dans la prédiction du modèle d'odométrie. De cette façon, l'estimation du filtre aura tendance à suivre de plus près les prédictions du modèle d'odométrie. Ce phénomène est visible sur la Figure 7.

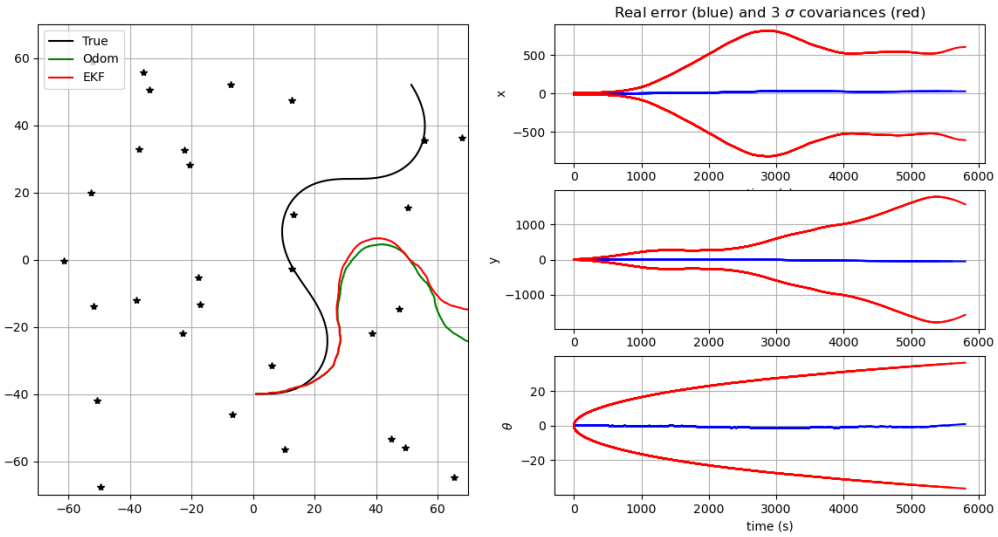

FIGURE 7 – Plot du code avec REst 10^7 fois plus grand que celui indiqué dans la Figure 1

Comme le montre la Figure 7, avec une erreur de mesure très importante, la prédiction du filtre EKF est plus proche de celle mesurée par le modèle d'odométrie. Par conséquent, le filtre devient plus sensible au bruit présent dans le modèle d'odométrie, ce qui présente une plus grande incertitude dans l'estimation.

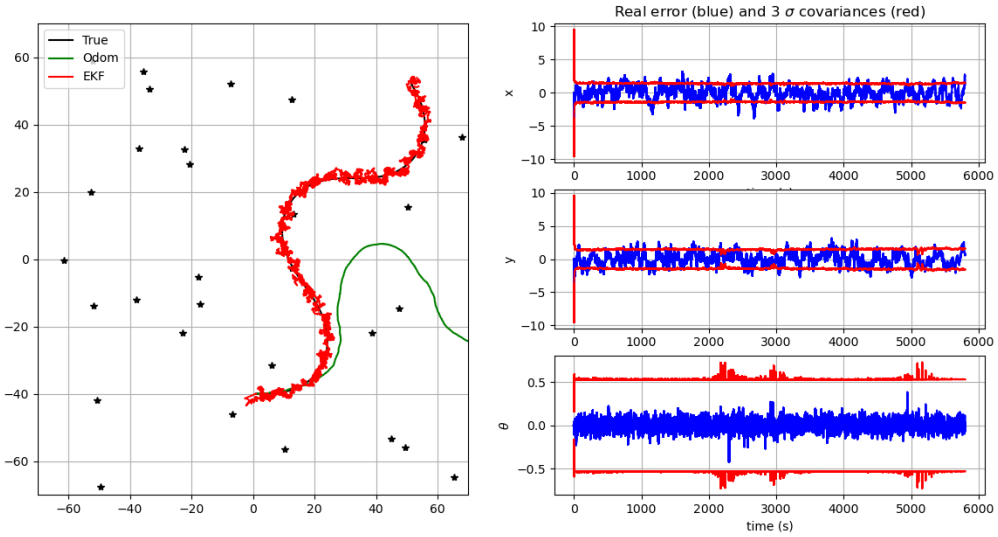

FIGURE 8 – Plot du code avec REst 100 fois plus petit que celui indiqué dans la Figure 1

En revanche, si l'erreur dans les mesures du filtre est très faible, le filtre accorde une grande confiance aux mesures et tient moins compte des prédictions du modèle odométrique. De cette manière, l'estimation du filtre de Kalman tend à suivre plus mesures que les prédictions de l'odométrie, ce qui rend le filtre de Kalman plus sensible aux mesures bruitées. L'erreur d'estimation de l'état du robot devient donc plus importante, ainsi que l'incertitude de l'état du robot. Cet effet est visible dans la Figure 8.

Exercice 6

Dans l'intervalle entre $t = 2500s$ et $t = 3500s$, aucun point de référence n'est observé. Le filtre ne dispose donc d'aucune mesure et ses prédictions sont basées uniquement sur les prédictions de l'odomètre. Par conséquent, le filtre a une grande incertitude dans l'estimation de l'état du robot, comme le montre la Figure 9. Comme on peut le voir, l'incertitude augmente à partir du moment où aucune mesure n'est prise ($t = 2500s$) et revient à son niveau précédent quand le filtre reçoit des observations de points de référence valides ($t = 2500s$).

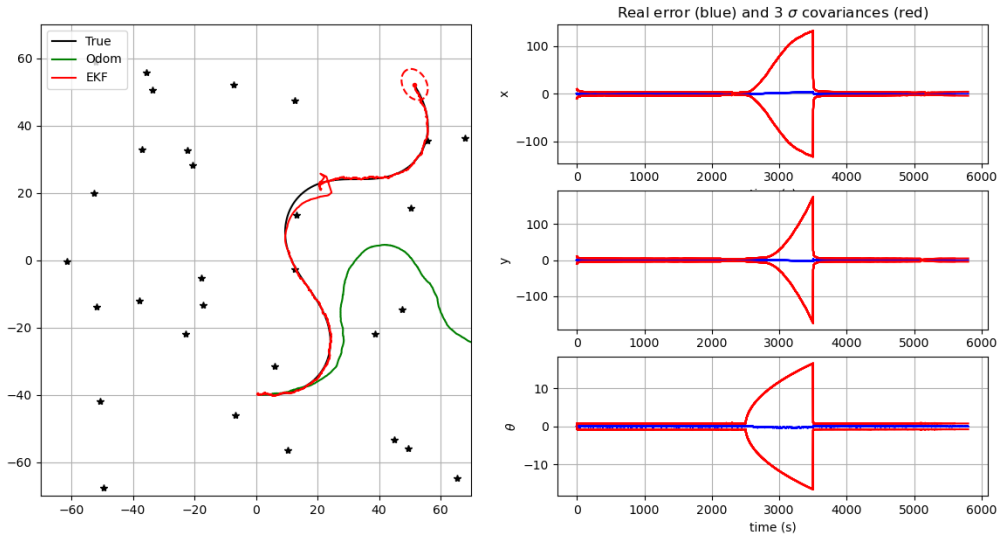

FIGURE 9 – Plot du code avec un trou de mesures entre $t = 2500s$ et $t = 3500s$.

De plus, on peut également voir sur la Figure 9 que dans l'intervalle de temps où le filtre se base uniquement sur les prédictions du modèle odométrique, la trajectoire prédite par le filtre de Kalman se rapproche de la trajectoire prédite par l'odomètre, puisqu'il n'y a pas d'étape de correction de la prédiction faite par le modèle odométrique à l'aide du gain de Kalman.

A l'instant $t > 3500s$, l'observation des points de référence redevient visible et le filtre de Kalman prend alors en compte à la fois les prédictions du modèle odomètre et les mesures obtenues à partir des points de référence. De cette manière, le filtre prédit à nouveau correctement la trajectoire du robot.

Exercice 7

En utilisant la Figure 1 comme référence, dans laquelle 30 amers ont été utilisés, nous pouvons voir qu'en réduisant le nombre de points à seulement 3 (Figure 10), la trajectoire prédite par le filtre reste similaire à celle avec 30 points. Cependant, une différence notable est la fréquence des chutes de l'incertitude au fil du temps. En raison de la présence de seulement 3 amers, et aucun d'entre eux n'étant proche de la trajectoire du robot, il n'y a pas de chute de l'incertitude (courbe rouge de la Figure 10).

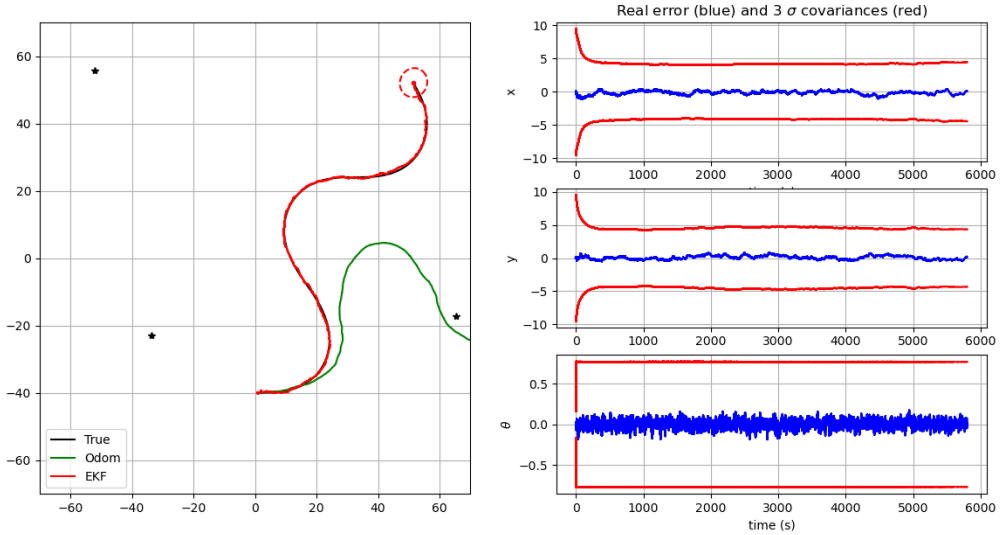

FIGURE 10 – Plot du code avec un nombre d'amers égal à 3.

En revanche, en augmentant considérablement le nombre de amers à 300 (Figure 11), la trajectoire prédite par le filtre reste similaire à celle avec 30 amers. De plus, en raison du plus grand nombre de amers, certains d'entre eux sont situés à proximité de la trajectoire du robot. De ce fait, le nombre de fortes baisses d'incertitude est plus important que celles observées avec 30 amers.

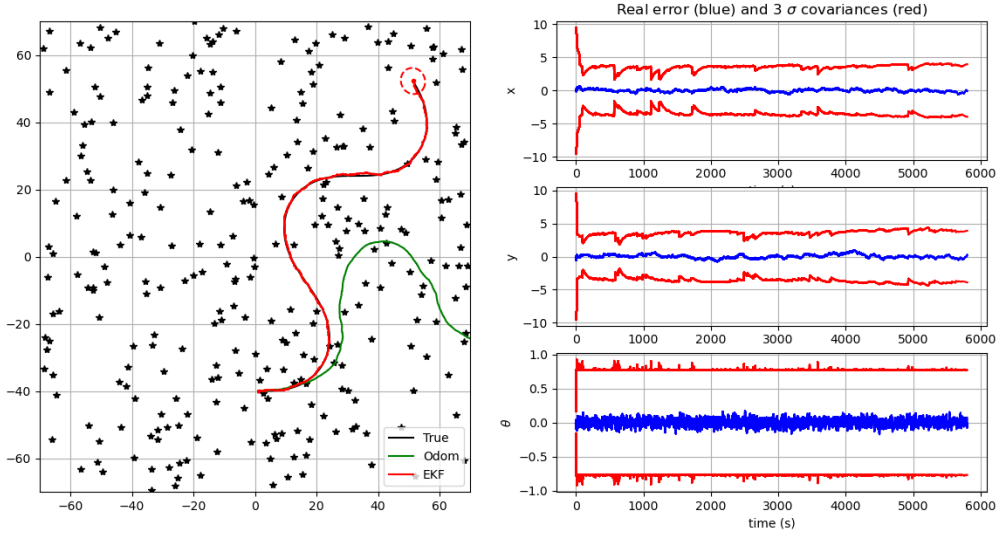

FIGURE 11 – Plot du code avec un nombre d'amers égal à 300.

On peut donc conclure que la variation du nombre d'amers n'affecte pas significativement les performances du filtre. Ce qui varie en revanche, c'est la fréquence des fortes baisses d'incertitude. En réalité, cette fréquence est plus liée à la présence d'amers proches de la trajectoire du robot qu'au nombre total d'amers. Un facteur qui a un impact réel sur les performances du filtre est la distribution des amers, et pas nécessairement leur nombre.

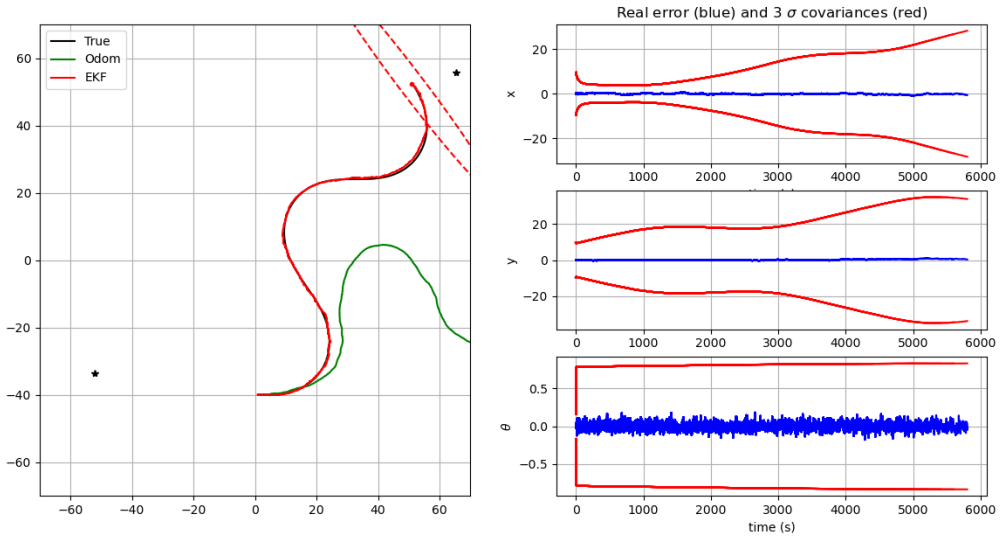

FIGURE 12 – Plot du code avec un nombre d'amers égal à 2.

La Figure 12 montre le cas où il n'y a que 2 amers et qu'ils sont très dispersés. Dans ce cas, comme les 2 amers sont très éloignés du robot et l'un de l'autre, la mesure de leur position a une très grande incertitude, ce qui compromet les performances du filtre. En conséquence, la prédiction du filtre devient plus imprécise, comme le montre l'augmentation de l'imprécision des états et la grande ellipse de la Figure 12.

Exercice 8

Dans le cas où seules des mesures de distance sont disponibles (*range only*), le filtre de Kalman implémenté donne les résultats suivants.

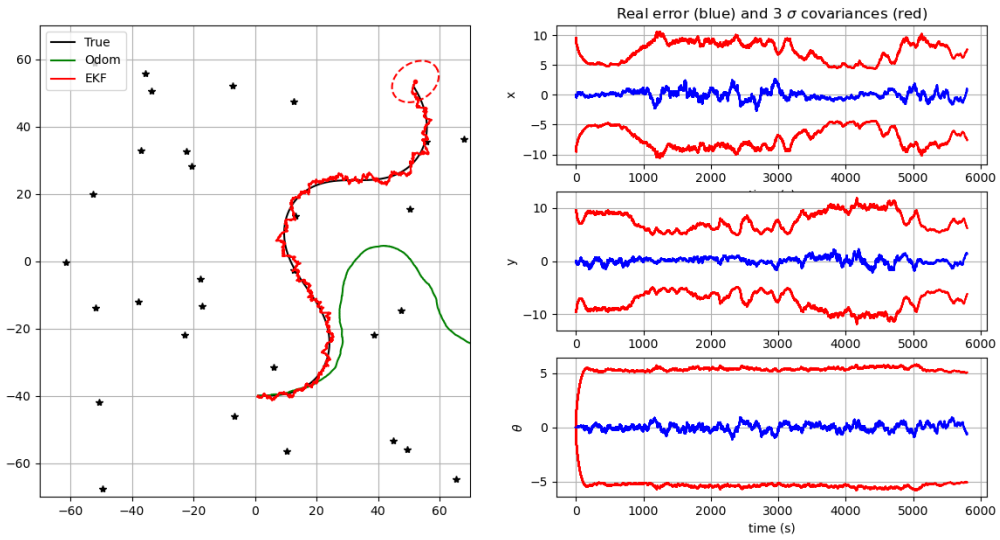

FIGURE 13 – Seulement les mesures de distance sont disponibles (*range only*).

Par rapport aux performances obtenues dans la Figure 1, où les directions sont également prises en compte, on constate une nette dégradation de la précision du filtre. La trajectoire prédite par le filtre de Kalman n'est plus précise par rapport à la trajectoire réelle, et il y a une variation notable de l'incertitude et des erreurs des prédictions.

Cette dégradation des performances est due à la perte d'informations angulaires sur le robot. Le filtre de Kalman perd sa capacité à déterminer la direction du robot par rapport aux points de référence, en se basant exclusivement sur la distance entre le robot et les points. En outre, comme le filtre se base uniquement sur la distance des points, plusieurs points peuvent se trouver à la même distance du robot, ce qui crée une ambiguïté dans la localisation du robot et la rend moins précise.

Même si on augmente le nombre d'amers, les performances ne s'amélioreront pas, car le plus important n'est pas le nombre d'amers, mais leurs distribution. C'est ce que montre la Figure 14. On peut donc conclure que la prise en compte des directions est fondamentale pour que le filtre de Kalman puisse prédire la trajectoire avec précision et obtenir de bonnes performances. Les informations angulaires jouent un rôle important dans la capacité du filtre à localiser avec précision le robot par rapport à des points de référence.

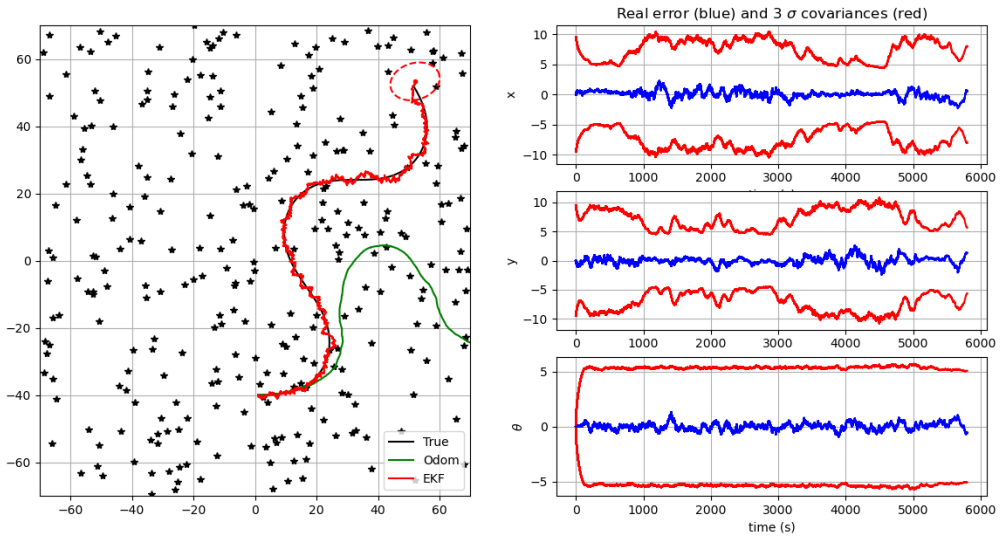

FIGURE 14 – Plot du code avec un nombre d'amers égal à 300.

Exercice 9

Dans le cas où seules des mesures de direction sont disponibles, le filtre donne des résultats différents de ceux obtenus dans le cas *range only*. Comme le montre la Figure 15, les performances sont aussi bonnes que celles obtenues dans le cas où les mesures de direction et de distance sont prises en compte (Figure 1).

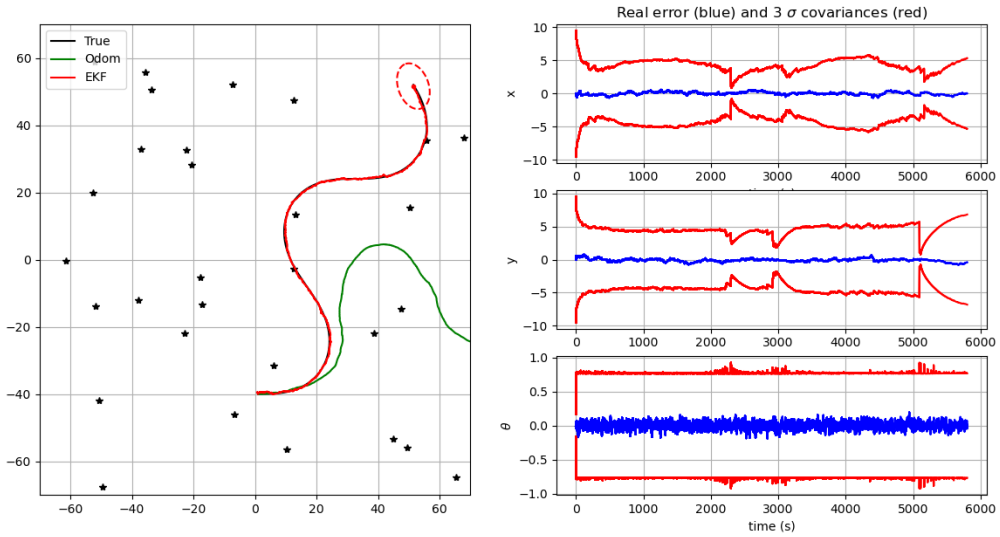

FIGURE 15 – Seulement les mesures de direction sont disponibles (*angles only*).

Comme le montre la Figure 15, la mesure de la direction (angle) est suffisante pour que le filtre de Kalman prédise correctement la trajectoire du robot. Même lorsque la distance entre le robot et les points d'intérêt est inconnue, le filtre de Kalman conserve un contrôle efficace sur l'orientation du robot. En outre, la mesure de la direction permet d'éviter les ambiguïtés dans la localisation des points de référence, puisque, avec la seule information angulaire, le filtre est capable d'établir une orientation précise du robot par rapport à ces points.

En augmentant le nombre d'amers (Figure 16), la performance ne change pas puisqu'un nombre suffisant de points garantit déjà une bonne performance du filtre, étant donné que la distribution des points est plus importante que leur nombre.

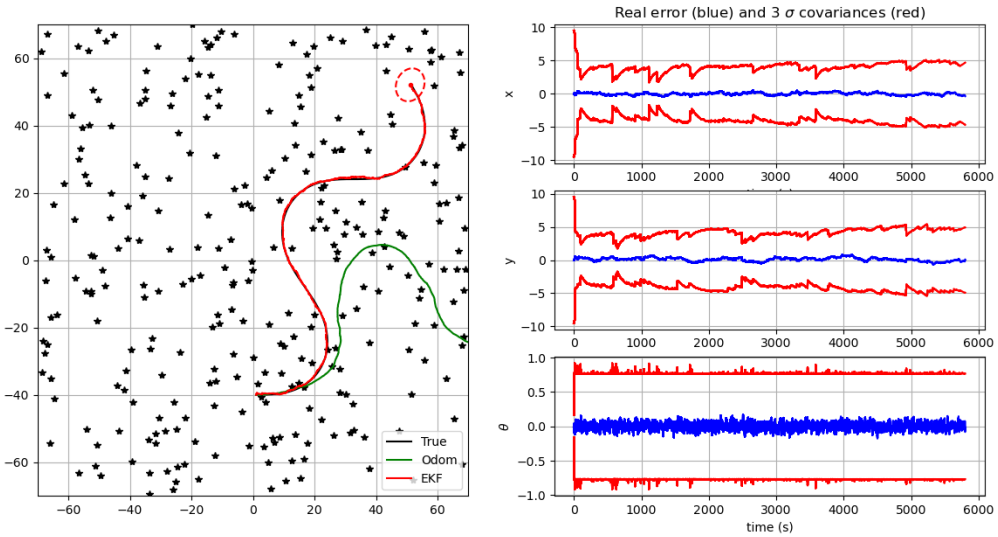

FIGURE 16 – Plot du code avec un nombre d'amers égal à 300.

En revanche, en réduisant le nombre de points de référence (Figure 17), le filtre parvient également à prédire la trajectoire du robot, mais avec une grande incertitude. En effet, les points sont mal répartis et leur nombre est insuffisant pour que le filtre puisse déterminer avec plus de certitude la localisation du robot.

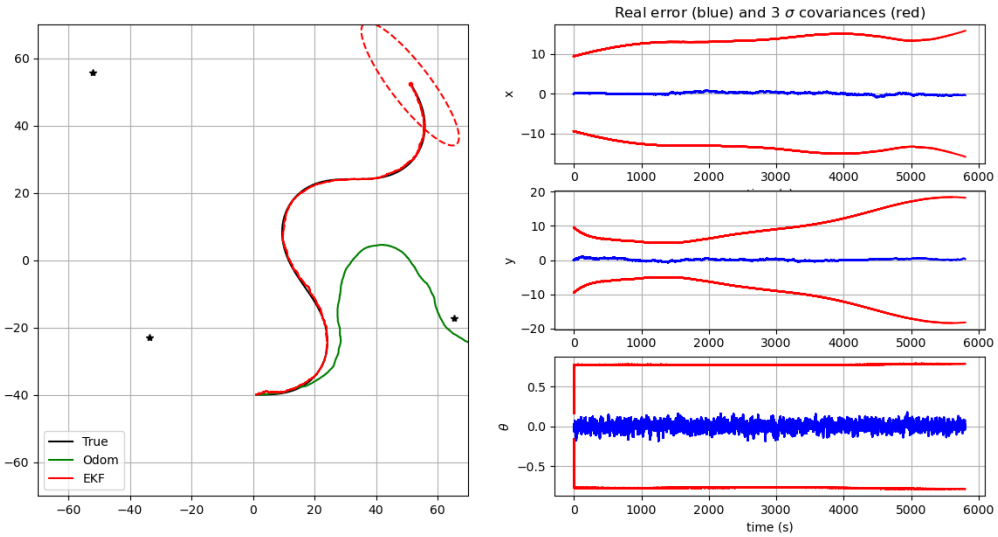

FIGURE 17 – Plot du code avec un nombre d'amers égal à 3.

Une solution consiste à augmenter le nombre de points de référence (Figure 16) ou à considérer les mesures de distance en plus des mesures de direction (Figure 10). Le filtre pourra alors prédire la trajectoire du robot avec précision et avec une faible incertitude.