

# TP4: Filtrage Particulaire

## Navigation pour les systèmes autonomes

### ROB312

Yu WANG

October 20, 2021

## 1 Introduction

Un filtre à particules est un filtre récursif qui utilise la méthode de Monte Carlo pour estimer l'état d'un système dynamique à partir d'une séquence d'observations bruyantes ou incomplètes en représentant la probabilité postérieure d'un événement aléatoire par un ensemble d'échantillons aléatoires (appelés particules) avec des poids. l'espace du modèle.

Le filtre à particules est une généralisation du filtre de Kalman, qui est basé sur un espace d'état linéaire et une distribution gaussienne du bruit, alors que le modèle d'espace d'état du filtre à particules peut être non linéaire et la distribution du bruit peut être de n'importe quel type. Par rapport aux modèles gaussiens, il peut exprimer une plus large gamme de distributions et possède une plus grande capacité à modéliser des caractéristiques non linéaires.

Malgré les nombreux avantages du filtrage particulaire, le grand nombre d'échantillons requis ne peut être ignoré : plus le nombre d'échantillons est important, plus l'algorithme est complexe. En outre, la phase de ré-échantillonnage entraîne une perte de validité et de diversité de l'échantillon.

## 2 Q1 - La structure de code et les paramètres

Cette section décrit la structure principale du code et la signification des paramètres. Afin de montrer plus clairement la structure du code, nous allons d'abord présenter la signification de quelques paramètres importants.

- **X\_reel** : l'état réel du système, puisqu'elle est inconnue, c'est aussi la valeur que nous voulons estimer. Il est initialisé par la valeur estimée et l'erreur d'initialisation

- **R** : la matrice de covariance du bruit de mesure réelle
- **T** : La durée de la simulation
- **N** : Le nombre de particules
- **P\_hat** : La matrice de covariance
- **X\_hat** : L'estimation initial ( $x, y, z, \theta$ )
- **Qf** : La matrice de covariance de bruit de dynamique
- **Rf** : La covariance du bruit de mesure du filtre
- **threshold\_resampling** : Le seuil de ré-échantillonnage ( $\theta_{eff}$ ). Quand  $N_{eff} = \frac{1}{\sum_i(w_k^i)^2} < \theta_{eff}N$ , on va faire le ré-échantillonnage.
- **Xp** : Le tirage des particules autour de **X\_hat** initial
- **wp** : Les poids initiaux associés aux particules
- **Y** : la simulation physique des mesures

Ayant compris la signification des paramètres ci-dessus, nous pouvons facilement comprendre la structure de l'ensemble du code (illustré à la Figure 1). Une fois que le temps de simulation a atteint  $T$ , nous déroulons la simulation et présentons ensuite les résultats.

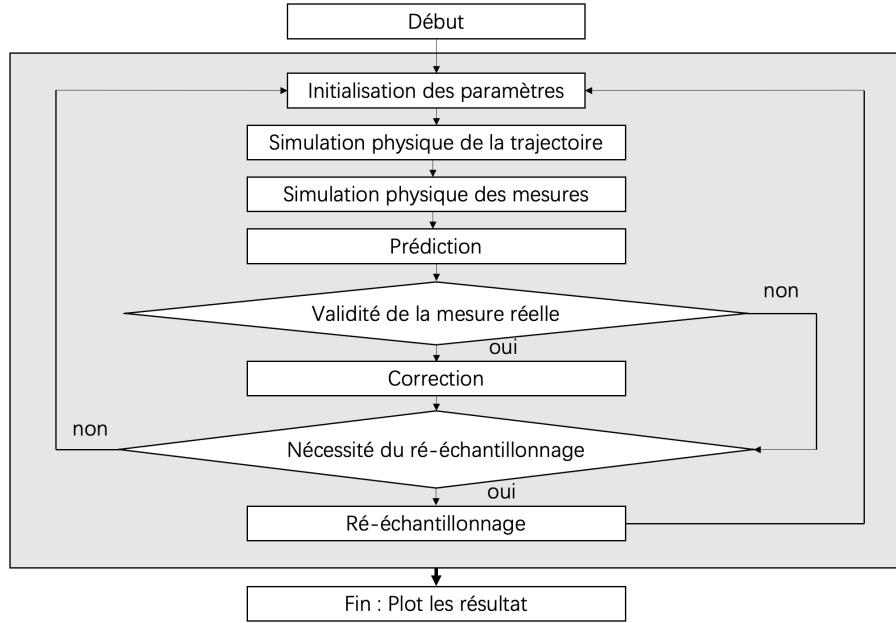
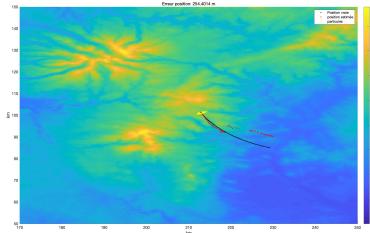


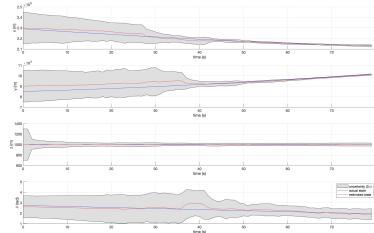
Figure 1: La structure de code

### 3 Q2 - Compléter le code et commenter le résultat

Sur la base de l'équation du filtre PF (prédiction, correction, ré-échantillonnage diapo 24) et le modèle dynamique et le modèle de mesure (diapo 33), on peut compléter le code. Les résultats sont présentés dans la Figure 2.



(a) La position vraie et estimée



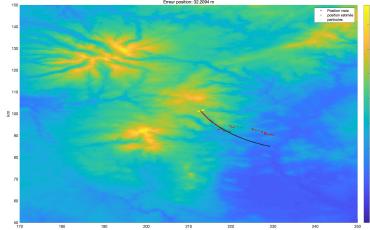
(b) Incertitude d'estimation

Figure 2: Résultat

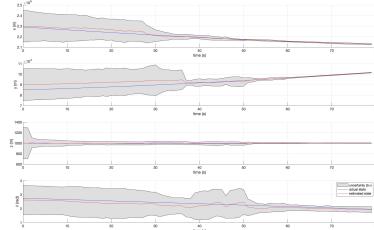
On peut constater que l'incertitude entre les états réels et estimés est également importante au début. Cependant, le filtrage particulier a permis d'estimer la position réelle et de faire converger les incertitudes. Finalement, l'erreur entre les états réel et estimé a été réduite à 458m et l'incertitude est beaucoup plus faible qu'au début.

### 4 Q3 - Faire varier le bruit de dynamique du filtre

Dans cette section, on fait varier le bruit de dynamique (matrice  $\mathbf{Qf}$ ) du filtre pour étudier l'effet du bruit dynamique du filtre sur son comportement. On a utilisé 2 valeurs :  $0.1\mathbf{Qf}$  et  $10\mathbf{Qf}$ .



(a) La position vraie et estimée



(b) Incertitude d'estimation

Figure 3: Résultat avec  $0.1\mathbf{Qf}$

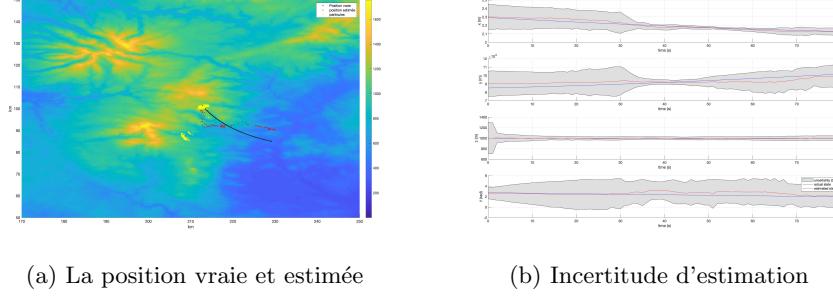


Figure 4: Résultat avec  $10Qf$

Le bruit dynamique ( $\mathbf{Qf}$ ) interfère avec la procédure de prédiction du filtre particulaire. Un bruit dynamique trop important peut empêcher la prédiction correcte des états, ce qui nous conduit alors à trop d'erreurs et d'incertitudes. Sur la base des observations des Figure 2, Figure 3 et Figure 4, on constate que plus le bruit dynamique est faible, plus l'incertitude de l'estimation est faible et plus l'erreur d'estimation finale est faible. Plus important encore, dans le cas de  $0.1Qf$ , l'erreur finale est de 32m, ce qui est moins que dans le cas de la figure 2, mais dans le cas de  $10Qf$ , l'erreur est de 2430m, ce qui est un écart trop grand.

## 5 Q4 - Faire varier le bruit de mesure du filtre

Dans cette section, on fait varier le bruit de mesure (matrice  $\mathbf{Rf}$ ) afin de déterminer l'effet du bruit de mesure du filtre sur son comportement. On a essayé différentes valeurs de  $\mathbf{Rf}$  entre  $10^2$  et  $100^2$  :  $10^2$ ,  $60^2$ ,  $100^2$  (par défaut  $\mathbf{Rf} = 20^2$ ). Tous les autres paramètres étaient définis par défaut.  $\mathbf{Rf}$  interfère avec la correction des poids des particules et affecte directement l'incertitude.

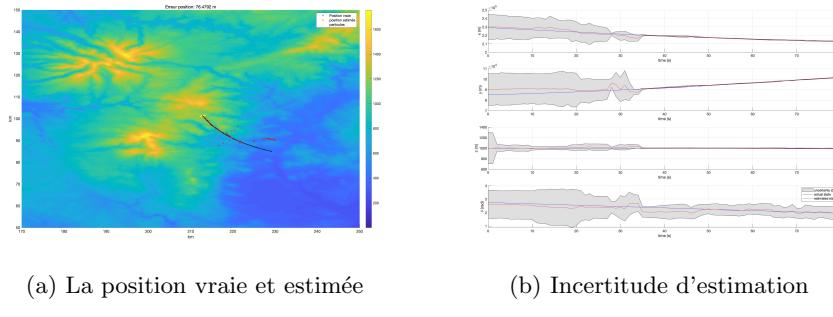


Figure 5: Résultat avec  $\mathbf{Rf} = 10^2$

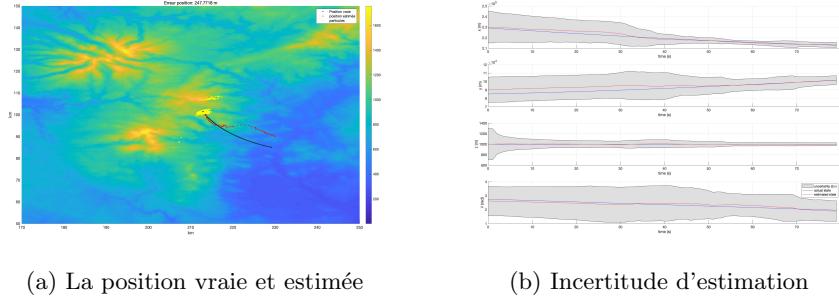


Figure 6: Résultat avec  $\mathbf{Rf} = 60^2$

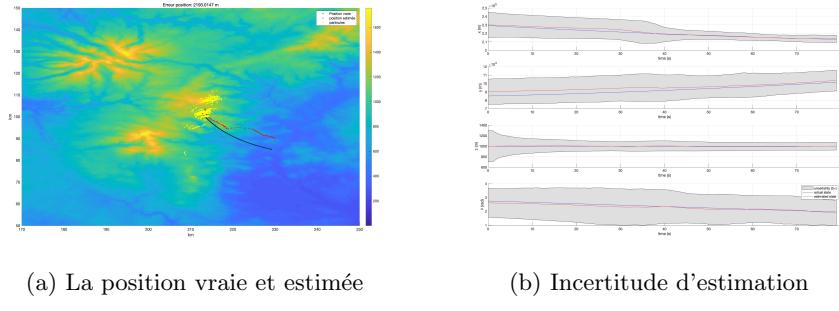


Figure 7: Résultat avec  $\mathbf{Rf} = 100^2$

Sur la base de Figure 5, Figure 6, Figure 7 et Figure 2, on constate que plus le bruit de mesure est petit, plus la convergence de l'incertitude se produit tôt, c'est-à-dire que il faut moins de temps pour que l'incertitude devienne plus petite. En outre, lorsque  $\mathbf{Rf}$  est trop grand, l'incertitude devient importante tout au long de la simulation.

## 6 Q5 - Faire varier le nombre de particules

Dans cette section, on va étudier l'effet du nombre de particules  $N$  sur le comportement du filtre en essayant différentes valeurs de  $N$  : 300, 30000 (par défaut  $N = 300$ ). Tous les autres paramètres étaient définis par défaut.

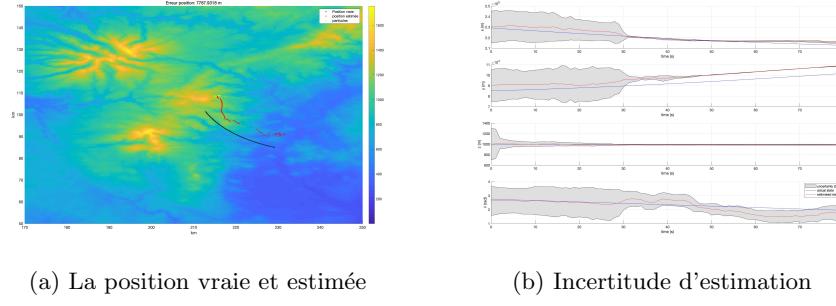


Figure 8: Résultat avec  $N = 300$

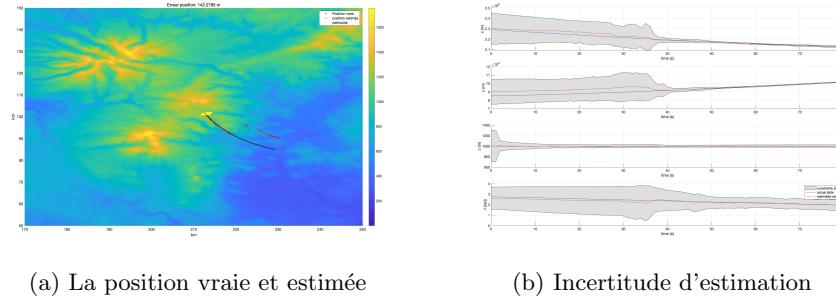
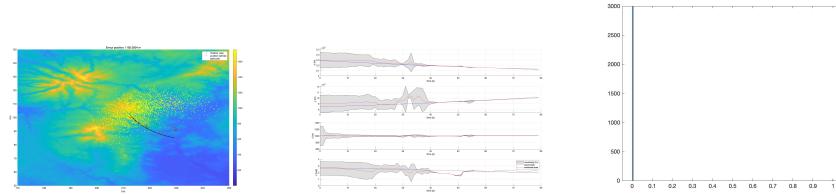


Figure 9: Résultat avec  $N = 300$

D'après Figure 8, Figure 9 et Figure 2, on constate que l'incertitude devient plus petite lorsqu'il y a moins de particules, mais l'erreur associée est plus grande et le filtre ne fonctionne pas bien. On peut en déduire que plus le nombre de particules est élevé, plus le calcul est lent et plus la convergence est bonne (plus petite l'erreur), mais plus le temps nécessaire pour réduire l'incertitude est long.

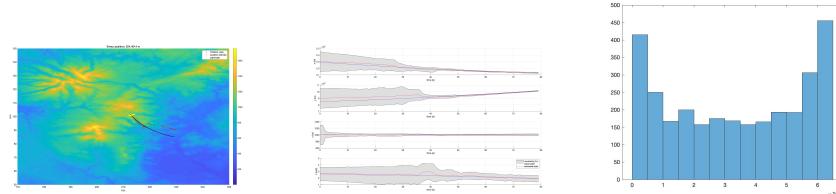
## 7 Q6 - Faire varier le seuil de ré-échantillonnage

Dans cette section, on va étudier l'effet des seuils de ré-échantillonnage sur le comportement du filtre en essayant différents seuils : 0, 0.5 et 1. Un histogramme des poids sera également tracé. Tous les autres paramètres étaient définis par défaut.



(a) La position vraie et es-  
(b) Incertitude d'estimation (c) Histogramme des poids  
timée

Figure 10: Résultat avec  $\theta_{eff} = 0$



(a) La position vraie et es-  
(b) Incertitude d'estimation (c) Histogramme des poids  
timée

Figure 11: Résultat avec  $\theta_{eff} = 0.5$

D'après Figure 10, Figure 11 et Figure 12, on constante que lorsque  $\theta_{eff} = 0$ , on ne fait jamais de ré-échantillonnage. L'histogramme basé sur le poids a des poids est 0. Lorsque  $\theta_{eff} = 0.5$  (par défaut), on ne fait jamais de ré-échantillonnage. Le ré-échantillonnage a eu lieu et le poids de chaque particule a changé, certaines particules ayant vu leur poids augmenter et d'autres ayant vu leur poids diminuer. Lorsque  $\theta_{eff} = 1$ , un plus grand ré-échantillonnage a été effectué (à chaque fois) et le problème de dégénérescence apparaît, car tous les poids des particules avaient des valeurs même (0.3333).

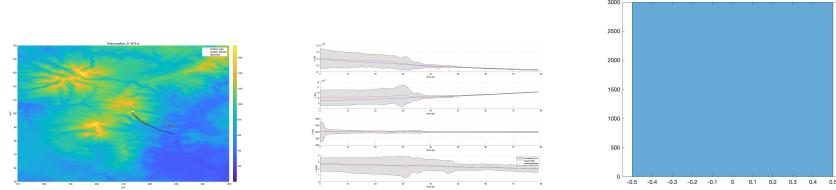
## 8 Q7 - Un trou de mesures entre t = 50s et t = 75s

Dans cette section, on va simuler un trou entre  $t = 50s$  et  $t = 75s$  en utilisant le code ci-dessous. Tous les autres paramètres étaient définis par défaut.

```

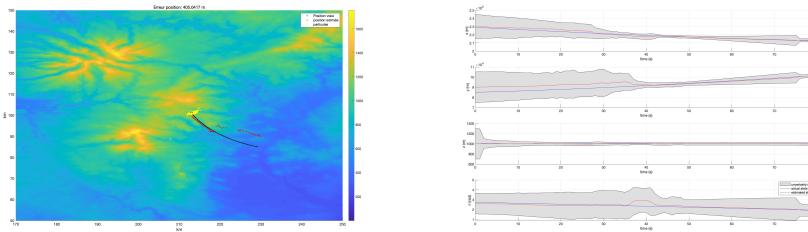
1 is_measurementValid = true;
2 if tk*dt >= 50 && tk*dt <= 75      % t >= 50s et t <= 75s
3     is_measurementValid = false;
4 end

```



(a) La position vraie et es-(b) Incertitude d'estimation (c) Histogramme des poids  
timée

Figure 12: Résultat avec  $\theta_{eff} = 1$



(a) La position vraie et estimée

(b) Incertitude d'estimation

Figure 13: Résultat avec un trou de mesures entre  $t = 50s$  et  $t = 75s$

Selon la Figure 13, on constante que l'incertitude augmente dans le trou à cause de manque de mesure et la correction des poids. Malgré le manque de mesures, le filtre final a tout de même été corrigé en temps après 75s et a finalement estimé l'état correctement, montrant une petite erreur de 405m.

## 9 Q8 - Modifier la fréquence des mesures

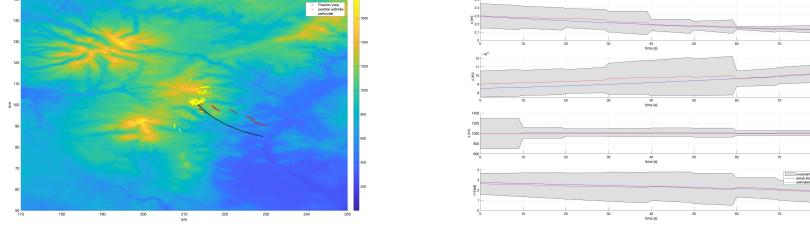
Dans cette section, on va simuler avec la fréquence de mesure ( $0.1Hz$ ) en utilisant le code ci-dessous. Tous les autres paramètres étaient définis par défaut.

```

1 is_measurementValid = true;
2 if mod(tk*dt, 10) ≠ 0
3     is_measurementValid = false;
4 end

```

Selon la Figure 14, on constante que l'incertitude diminue toutes les 10 secondes en utilisant la fréquence de mesure =  $0.1Hz$ . Par rapport à la Figure 2 où les mesures sont toujours disponibles, cela nous donne une image plus claire de la façon dont les mesures nous aident à réduire l'incertitude et de la façon dont l'incertitude augmente lorsqu'il n'y a pas de mesures.



(a) La position vraie et estimée

(b) Incertitude d'estimation

Figure 14: Résultat avec la fréquence de mesure =  $0.1\text{Hz}$ 

## 10 Q9 - Une autre façon de ré-échantillonner les poids Li et al. (2015)

Dans cette section, on va utiliser une méthode alternative de ré-échantillonnage des poids. L'algorithme de ré-échantillonnage polynomial ayant été utilisé dans le code, on choisi donc d'utiliser la méthode de ré-échantillonnage stratifié, dont l'algorithme est illustré à la Figure 15.

**Code 3:** Multinomial/stratified/systematic resampling.

---

```

 $[\{\tilde{x}_t^{(n)}\}_{n=1}^N] = \text{Resample}[\{x_t^{(m)}, w_t^{(m)}\}_{m=1}^M, N]$ 

 $[\{Q_t^{(m)}\}_{m=1}^M] = \text{CumulativeSum}[\{w_t^{(m)}\}_{m=1}^M]$ 
 $n = 0$ 
/Systematic/stratified choice runs:
 $m = 1$ 
/Systematic choice runs
 $u_0 \sim U(0, 1/N)$ 
WHILE ( $n \leq N$ )
/Stratified choice runs
 $u_0 \sim U(0, 1/N]$ 
/Systematic/stratified choice runs
 $u = u_0 + n/N$ 
/Multinomial choice runs
 $u \sim U(0, 1]; m = 1$ 
WHILE ( $Q_t^{(m)} < u$ )
 $m = m + 1$ 
END
 $n = n + 1$ 
 $\tilde{x}_t^{(n)} = x_t^{(m)}$ 
END

```

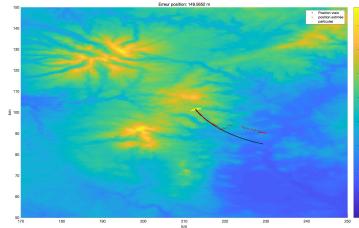
---

Figure 15: Algorithme de ré-échantillonner les poids

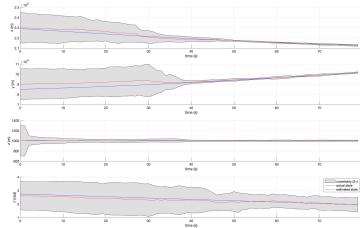
```

1 function [index] = stratified(wp)
2     N = length(wp);
3     index = zeros(1,N);
4     Q = cumsum(wp);
5     n = 1;
6     m = 1;
7     while (n <= N && m <= N)
8         u0 = rand/N;
9         u = u0 + n/N;
10        while (Q(m) < u && m < N)
11            m = m+1;
12        end
13        index(n) = m;
14        n = n+1;
15    end
16 end

```



(a) La position vraie et estimée



(b) Incertitude d'estimation

Figure 16: Résultat avec la méthode de ré-échantillonnage stratifié

Les résultats de la méthode de ré-échantillonnage stratifié sont présentés à la Figure 16, où le filtre particulaire converge avec succès pour nous en termes d'erreur et d'incertitude. Son erreur finale n'est pas très différente de celle de la méthode polynomiale.

## References

- Li, T., Bolic, M., and Djuric, P. M. (2015). Resampling methods for particle filtering: Classification, implementation, and strategies. *IEEE Signal Processing Magazine*, 32(3):70–86.