Filtrage Particulaire appliqué à la localisation

Karim Dahia, Nicolas Merlinge

IA712

ONERA

THE FRENCH AEROSPACE LAB

ENSTA

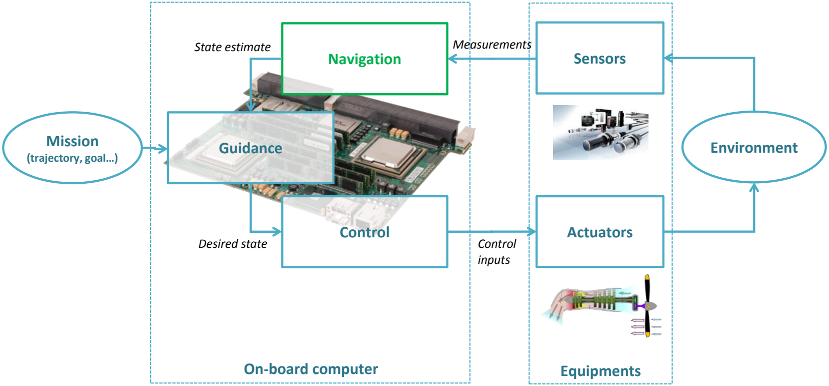
IP PARIS

Motivation

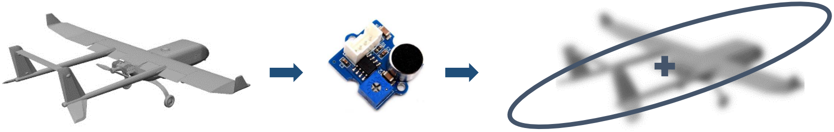
Autonomous systems need to perform accurate state estimation (e.g., self localization, objective assessment...)

... with limited embedded calculation capabilities.

Guidance, Navigation and Control (GNC)



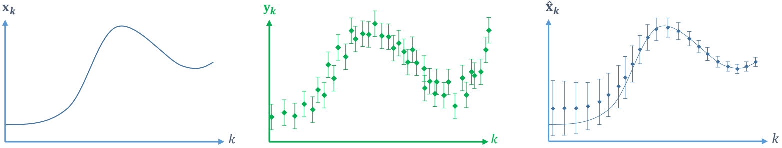
State estimation for navigation



$\mathbf{x}_k =$ position, velocity...

$\mathbf{y}_k =$ measurements

$\hat{\mathbf{x}}_k =$ estimated position, velocity...



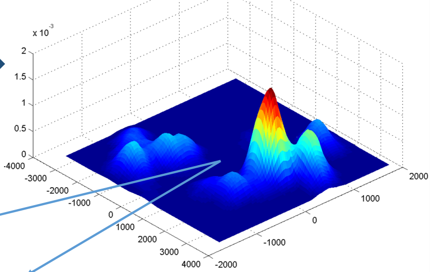
State estimation consists in retrieving the vehicle's state from noisy and incomplete measurements and uncertain evolution model.

Different ways to model the state density

$x_k =$ position, velocity...

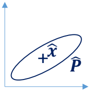
$y_k =$ measurement

Density: $p(x_k|y_1, \dots, y_k)$

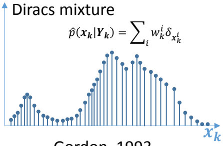


Kalman Filters

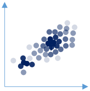
Particle Filters



Kalman, 1960



Gordon, 1993



What we know

Theoretical state evolution (dynamical model):

$$\begin{aligned} \dot{\mathbf{x}} &= F(\mathbf{x}, \mathbf{u}) \\ \mathbf{x}_k &= f(\mathbf{x}_{k-1}, \mathbf{u}_k) \end{aligned} \tag{1}$$

Theoretical observation equation (sensor model):

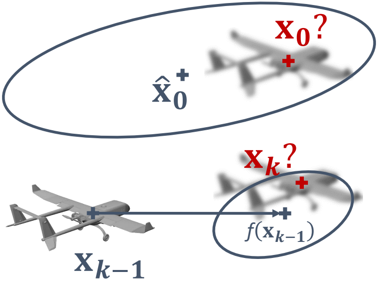
$$\mathbf{y}_k = h(\mathbf{x}_k) \tag{2}$$

However, these equations are not totally representative of the actual system, e.g.:

- unexpected wind, friction, unmodeled dynamics...
- sensor noise, unmodeled disturbances...

What we **don't** know (uncertainties)

- Initial state uncertainty

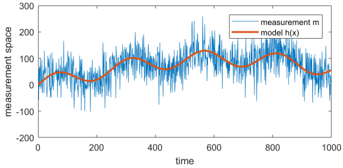


- Process noise (dynamics)

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_k) + \mathbf{w}_k$$

- Measurement noise (and potentially some bias)

$$\mathbf{y}_k = h(\mathbf{x}_k) + \mathbf{v}_k$$



A way to model uncertainties: probability distribution functions

- State distribution:

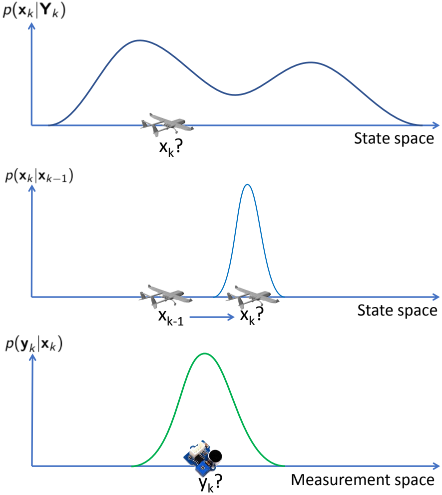
$$p(\mathbf{x}_k|\mathbf{y}_1,\ldots,\mathbf{y}_k) \triangleq p(\mathbf{x}_k|\mathbf{Y}_k)$$

- Process noise distribution

$$p(\mathbf{x}_k|\mathbf{x}_{k-1})$$

- Measurement distribution

$$p(\mathbf{y}_k|\mathbf{x}_k)$$



Optimal filter equations (Bayesian filtering)

State density propagation (dynamics, Chapman-Kolmogorov equation):

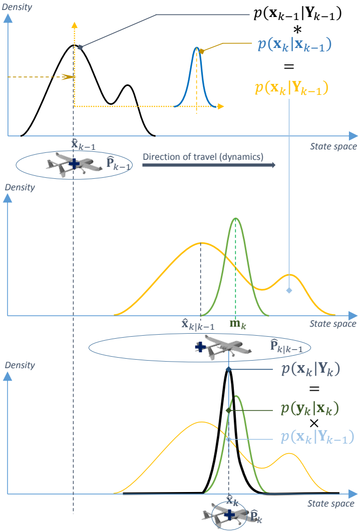
$$p(\mathbf{x}_k|\mathbf{Y}_{k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})\mathrm{d}\mathbf{x}_{k-1} \tag{3}$$

State density correction/update (measurements, Bayes rule):

$$p(\mathbf{x}_k|\mathbf{Y}_k) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{Y}_{k-1})}{\int p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{Y}_{k-1})\mathrm{d}\mathbf{x}_k} \tag{4}$$

(Measurements $\mathbf{y}_i$ and $\mathbf{y}_j$ ($\forall\ i \neq j$) are assumed to be statistically independent)

Optimal filter equations (Bayesian filtering)



(a) Prediction

Convolution of the **prior conditional density** with the state transition density. The transition density accounts for the deterministic dynamics f_k and its uncertainty (process noise $\mathbf{w}_k$).

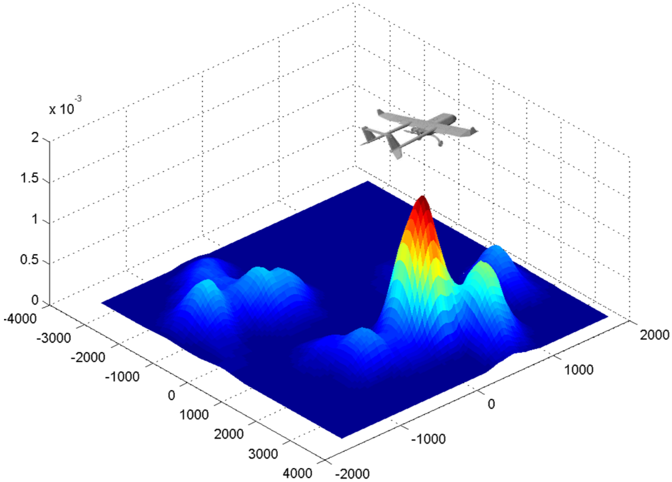
(b) Predicted density, new measurement

Step (a) results in the **predicted conditional density**, whose support is usually larger than the prior density.
A measurement $\mathbf{y}_k$ is now available. It will introduce information in the estimation.

(c) Correction

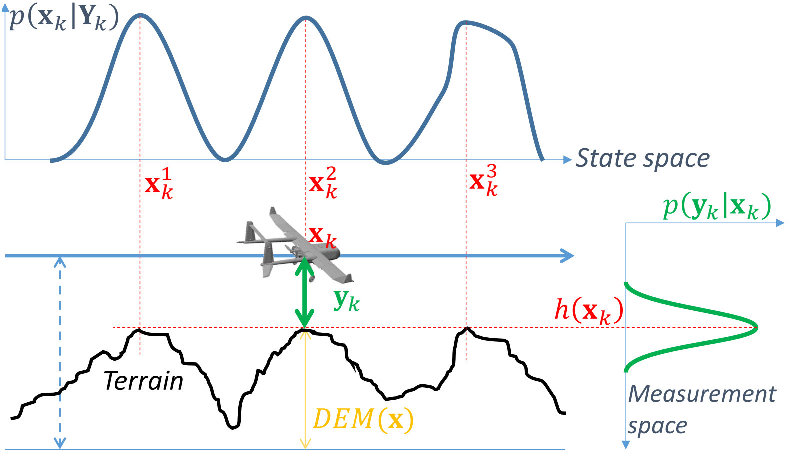
The predicted density is multiplied with the measurement density, leading to the **posterior conditional density**.
Its support is usually smaller than the predicted density. It yields a refined estimate $\hat{\mathbf{x}}_k$ and covariance $\hat{\mathbf{P}}_k$.
Then, one can iterate back to step (a) for further time-steps.

Multimodal state density (example: Terrain Aided Navigation)



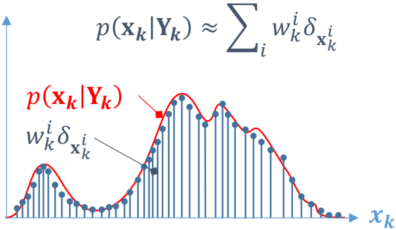
State density $p(\mathbf{x}_k|\mathbf{Y}_k)$

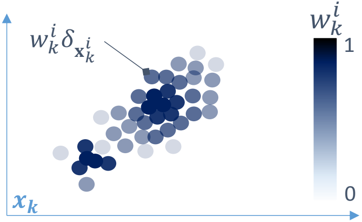
Multimodal state density (example: Terrain Aided Navigation)



Non injective observation equation (e.g. terrain elevation) may yield several admissible states (peaks in posterior state density).

State density approximation





$$p(\mathbf{x}_k|\mathbf{Y}_k) \approx \sum_{i=1}^{N} w_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i) \tag{5}$$

$\mathbf{x}_k$: actual state
$\mathbf{x}_k^i$: particle i state
w_k^i : particle i weight $(\sum_i w_k^i = 1)$

Particle Filter (theory): prediction and correction

State density propagation (dynamics, Chapman-Kolmogorov equation):

$$\begin{array}{ccc} p(\mathbf{x}_k|\mathbf{Y}_{k-1}) & = \int & p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{Y}_{k-1})\mathrm{d}\mathbf{x}_{k-1} \\ \sum_{i=1}^N w_{k-1}^i \delta(\mathbf{x}_k - \mathbf{x}_k^i) & \leftarrow & \sum_{i=1}^N w_{k-1}^i \delta(\mathbf{x}_k - (f(\mathbf{x}_{k-1}^i, \mathbf{u}_k) + \mathbf{v}_k^i)) \end{array} \tag{6}$$

State density correction/update (measurements, Bayes rule):

$$\begin{array}{ccc} p(\mathbf{x}_k|\mathbf{Y}_k) & \propto & p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{Y}_{k-1}) \\ \sum_{i=1}^N w_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i) & \leftarrow & \sum_{i=1}^N w_{k-1}^i p(\mathbf{y}_k|\mathbf{x}_k^i)\delta(\mathbf{x}_k - \mathbf{x}_k^i) \end{array} \tag{7}$$

Particle Filter (in practice): prediction and correction

State density propagation (dynamics, Chapman-Kolmogorov equation):

$$\mathbf{x}_k^i = f(\mathbf{x}_{k-1}^i, \mathbf{u}_k) + \mathbf{w}_k^i \tag{8}$$

$\mathbf{w}_k^i \ \forall i \in [1, N]$: random sample of process noise

State density correction/update (measurements, Bayes rule):

$$w_k^i \propto w_{k-1}^i p(\mathbf{y}_k | \mathbf{x}_k^i) \tag{9}$$

Particle Filter: state estimation

Maximum *a posteriori* (hard to compute in the general case):

$$\widehat{\mathbf{x}}_k = \underset{\mathbf{x}_k}{\operatorname{argmax}}\, p(\mathbf{x}_k|\mathbf{Y}_k) \tag{10}$$

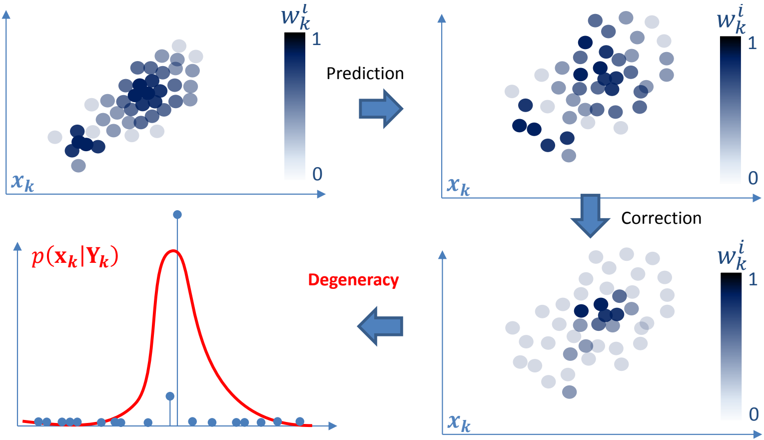
Least square approximation:

$$\widehat{\mathbf{x}}_k = \sum_{i=1}^{N} w_k^i \mathbf{x}_k^i \tag{11}$$

Empirical covariance:

$$\widehat{\mathbf{P}}_k = \sum_{i=1}^{N} w_k^i (\mathbf{x}_k^i - \widehat{\mathbf{x}}_k)(\mathbf{x}_k^i - \widehat{\mathbf{x}}_k)^T \tag{12}$$

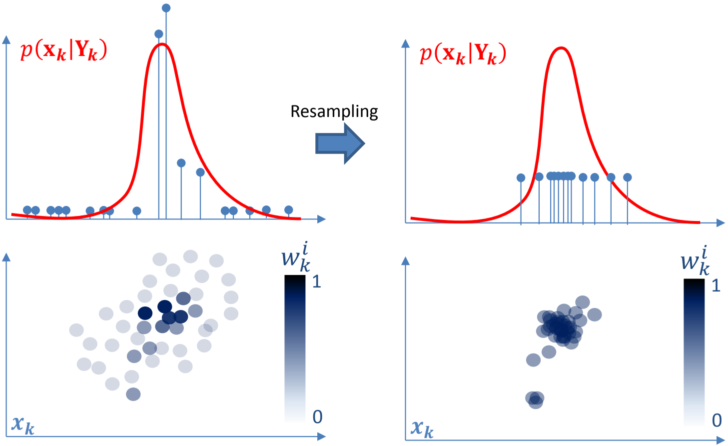
Prediction, correction, and degeneracy phenomenon



After a number of [prediction-correction] cycles, a majority of particle weights will tend to 0 while a small number (or only one) will tend to 1:

That is the weights degeneracy phenomenon.

Resampling



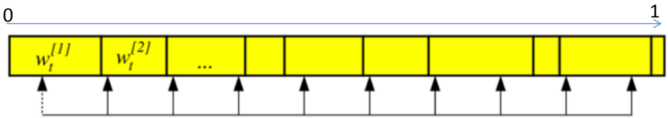
The resampling step aims to duplicate strong-weighted particles to keep an appropriate description of the state density **when degeneracy is about to occur**. Low-weighted particles are destroyed to keep N unchanged.

Multinomial Resampling

The most commonly used resampling technique is Multinomial Resampling:

Input: particle weights $\{w^i\}_{i\in[1,N]}$
Output: number of new instances per particles $\{n^i\}_{i\in[1,N]}$

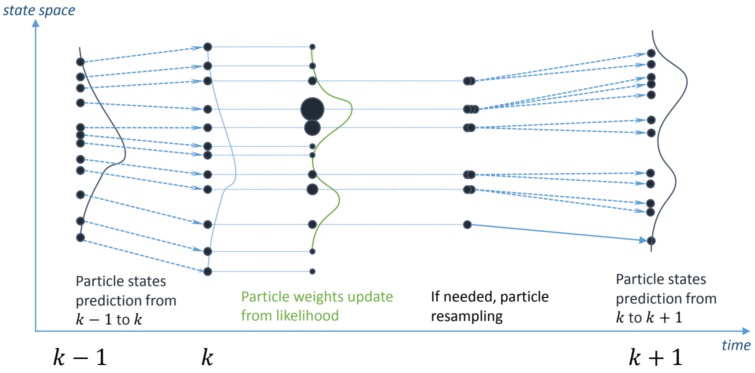
Initialise the duplication counters to $n^i = 0\ \forall i \in [1, N]$
for $i = 1$ to N **do**
  Draw $u^i \sim \mathcal{U}_{[0,1]}$
  Find $j \in [1, N]$ such that $u^i \in \left]\sum_{l=1}^{j-1} w^l, \sum_{l=1}^{j} w^l\right]$
  Count $n^j = n^j + 1$
end for
Return $n^i\ \forall i \in [1, N]$



for $i = 1$ to N **do**
Draw $u^i \sim \mathcal{U}_{[0,1]}$

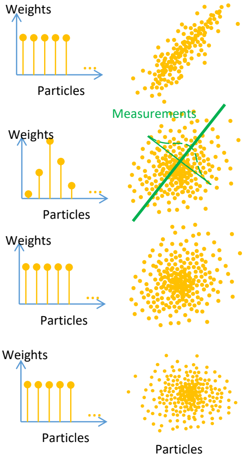
New particle distribution

Prediction, correction, and resampling



SIR-PF scheme

Particle Filter



Steps

Resampling

Low-weighted particles are replaced with copies of high-weighted particles

Correction

$$\begin{cases} p(\mathbf{x}_k|\mathbf{Y}_k) = \sum_{i=1}^{N} \omega_k^i \delta_{\mathbf{x}_k^i} \\ \omega_k^i \propto p\left(\mathbf{y}_k|\mathbf{x}_k^i\right)\omega_{k-1}^i \end{cases}$$

Prediction

$$\begin{cases} p(\mathbf{x}_k|\mathbf{Y}_{k-1}) = \sum_{i=1}^{N} \omega_{k-1}^i \delta_{\mathbf{x}_k^i} \\ \mathbf{x}_k^i = f\left(\mathbf{x}_{k-1}^i, \mathbf{u}_k\right) + \mathbf{w}_k^i \end{cases}$$

Initialization

$$\begin{cases} p(\mathbf{x}_0) = \sum_{i=1}^{N} \omega_0^i \delta_{\mathbf{x}_0^i} \\ \left\{\mathbf{x}_0^i\right\}_{i=\{1,\ldots,N\}} \sim p(\mathbf{x}_0) \end{cases}$$

PF formulation example with Gaussian uncertainties

Initialization: draw:

$$\left\{ \mathbf{x}_0^i \sim \mathcal{N}(\widehat{\mathbf{x}_0}, \widehat{\mathbf{P}}_0) \right\}_{i \in [1,N]} \tag{13}$$

Prediction step:

$$\mathbf{x}_k^i = f(\mathbf{x}_{k-1}^i, \mathbf{u}_k) + \mathbf{w}_k^i \tag{14}$$

where $\mathbf{w}_k^i \sim \mathcal{N}(0, \mathbf{Q}_k) \quad \forall i \in [1, N]$

Correction step:

Weights update (where $\mathbf{y}_k - h(\mathbf{x}_k^i)$ is the *innovation* term):

$$\tilde{w}_k^i = w_{k-1}^i \exp\left(-\frac{1}{2} (\mathbf{y}_k - h(\mathbf{x}_k^i))^T \mathbf{R}_k^{-1} ((\mathbf{y}_k - h(\mathbf{x}_k^i)) \right) \tag{15}$$

Weights normalization: $w_k^i = \frac{\tilde{w}_k^i}{\sum_i \tilde{w}_k^i}$

Estimation:

$$\widehat{\mathbf{x}_k} = \sum_{i=1}^{N} w_k^i \mathbf{x}_k^i \qquad \widehat{\mathbf{P}}_k = \sum_{i=1}^{N} w_k^i (\mathbf{x}_k^i - \widehat{\mathbf{x}}_k)(\mathbf{x}_k^i - \widehat{\mathbf{x}}_k)^T \tag{16}$$

Resampling (cf slide 19) if:

$$N_{eff} = \frac{1}{\sum_i (w_k^i)^2} < \theta_{eff} N \tag{17}$$

where $\theta_{eff} \in (0, 1)$, (usually 0.5)

Adaptive particle filters

KLD-sampling

At each iteration of the particle filter, determine the number of samples such that, with probability $1-\delta$, the error between the true posterior and the sample-based approximation is less than ε.

$$N = \frac{k-1}{2\varepsilon}\left[1 - \frac{2}{9(k-1)} + \sqrt{\frac{2}{9(k-1)}}z_{1-\delta}\right]^3 \qquad (18)$$

where $z_{1-\delta}$ is the upper $1-\delta$ quantile of the standard normal $\mathcal{N}(0,1)$ distribution and k is the numbers of bins.

Typical parameter values: $\varepsilon = 0.07$ and $1-\delta = 0.99$.

KLD: Kullback-Leibler Divergence.

KLD-sampling algorithm

Initialisation

NX=0; N=0; k=0 ;

Paramètres de réglage

ε = 0.07;

NMIN= 500 ;

Bins = [0,...,0]; les cases de la grille

While *N<NX ou N< NMIN*

N=N+1;

Tirage aléatoire de l'indice p de la particule

$\{X_{k-1}^p, w_{k-1}^p\}$

Prédiction

$X_k^p = f(X_{k-1}^p) + W_k^p$

Correction

$w_k^p \propto w_{k-1}^p p(y_k | X_k^p)$

Recherche du numéro de la case j occupée de la grille par la particule X_k^p

j = find(X_k^p *)*

Si bins(j) < 1

k = k+1; % incrémentation du nombre de case occupée par les particules

bins(j) = bins (j)+1; % la case est occupée

Si k>= 1

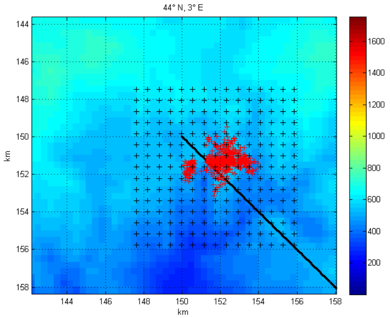
$$NX = \frac{k-1}{2\varepsilon}\left[1 - \frac{2}{9(k-1)} + \sqrt{\frac{2}{9(k-1)}}\, z_{1-\delta}\right]^3$$

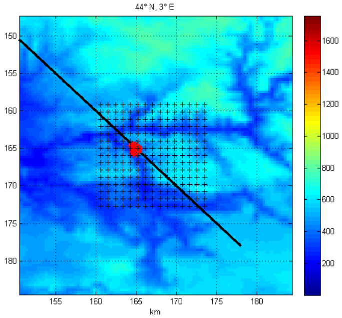
end

end

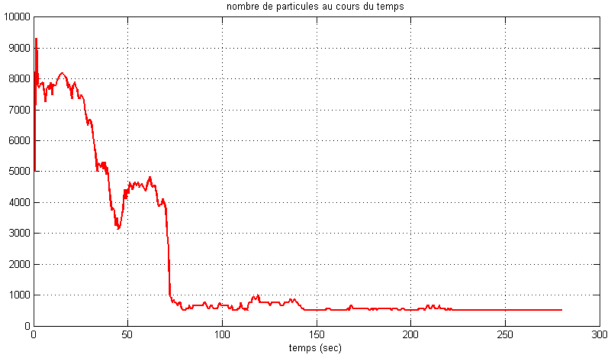
end

KLD-sampling

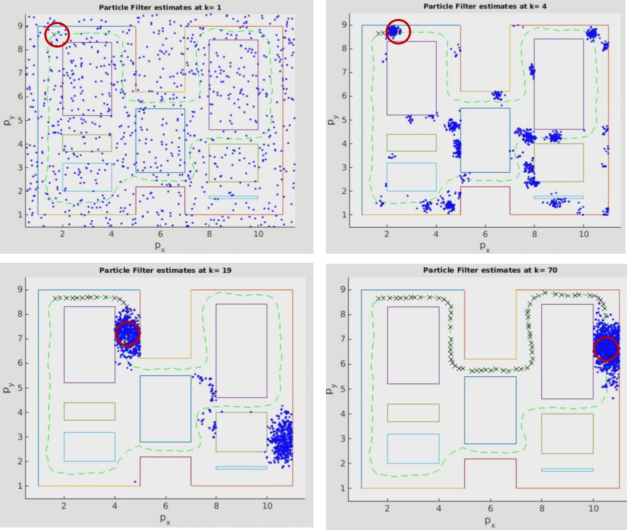




KLD-sampling

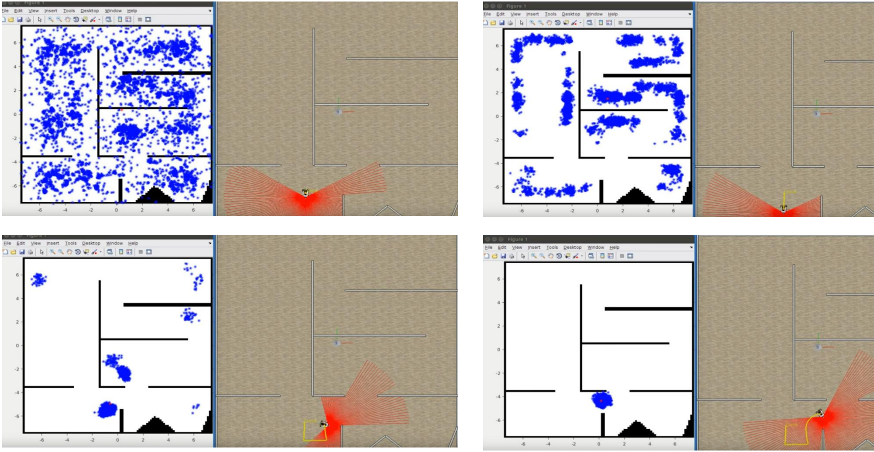


Ex1: Indoor map navigation (odometry only)



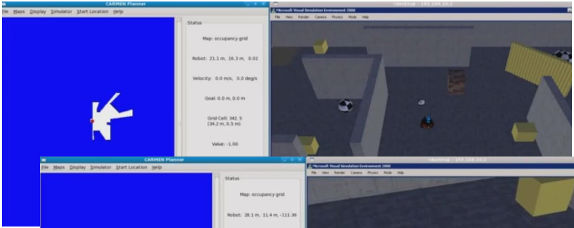
Position estimation using Particle Filter
https://www.youtube.com/watch?v=qsNGoHi7o2U

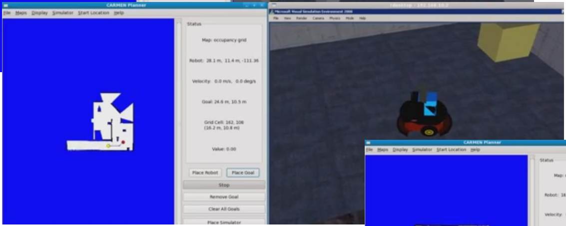
Ex2: Indoor map navigation with LIDAR data

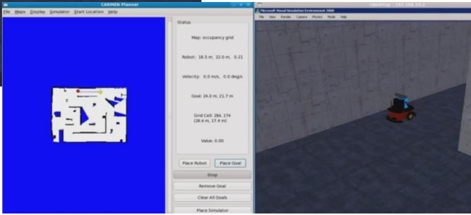


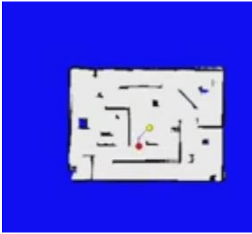
EMOR - Project 3: particle filter (with LIDAR data)
https://www.youtube.com/watch?v=Xlt1aoz5fUw

Ex3: Particle Filter SLAM



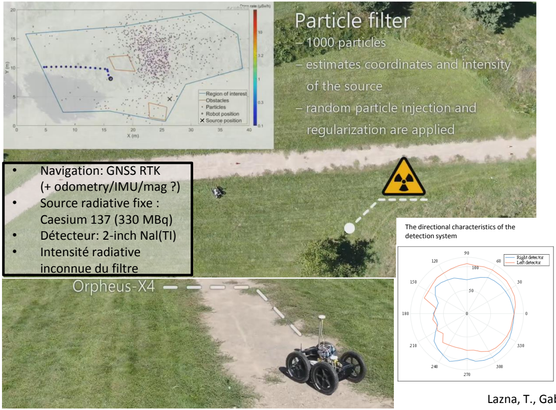


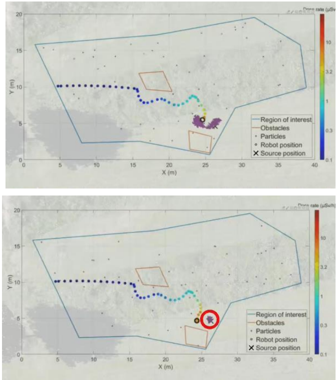




Particle Filter SLAM with Occupancy Grid Map
https://www.youtube.com/watch?v=CWpwrpZD0J4

Ex4: Autonomous Radiation Source Localization Using a Particle Filter

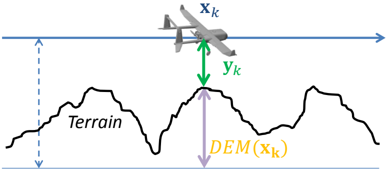


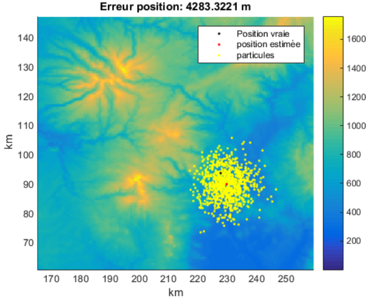


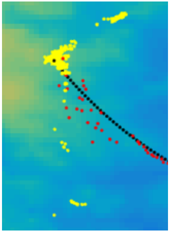
Autonomous Radiation Source Localization Using a Particle Filter
https://www.youtube.com/watch?v=Ox4J_Jov2XE

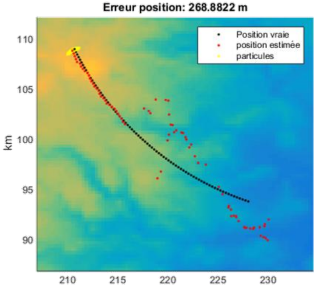
Lazna, T., Gabrlik, P., Jilek, T., & Zalud, L. (2018). Cooperation between an unmanned aerial vehicle and an unmanned ground vehicle in highly accurate localization of gamma radiation hotspots. *International Journal of Advanced Robotic Systems*, *15*(1), 1729881417750787.
https://journals.sagepub.com/doi/pdf/10.1177/1729881417750787

Ex5: Terrain aided navigation









SIR-PF advantages and limitations

Advantages

- Tackles non-linear dynamics and measurement models
- Tackles non-differentiable models (no linearization)
- Tackles non-Gaussian uncertainties and multimodal state densities (ambiguities)
- Can start with huge initial uncertainty
- Relatively easy to implement

Limitations of the original version of the algorithm:

- Computationally demanding
- May converge to erroneous local state density maximum
- Can be hard to tune (number of particles, choice of noise modeling, resampling threshold)

More advanced Particle Filters

Original algorithm:

- Sequential Importance Resampling Particle Filter (SIR-PF) [2]

More advanced algorithms:

- Rao-Blackwellized Particle Filter (RBPF) [7]
- (Weighted) Ensemble Kalman Filter (WEnKF/EnKF) [3]
- Regularized Particle Filter (RPF) [4]
- Kalman Particle Kernel Filter (KPKF) [6]
- Box Particle Filter (BPF) [8] and Box Regularized Particle Filter (BRPF) [9]
- Adaptive Approximate Bayesian Computational Particle Filter (A2BC-PF) [10]
- ...

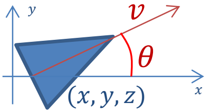
TP

1. Prendre en main et décrire sous forme de schéma la structure du code *FiltrageParticulaire_terrainNavigation.m* et repérer les différents paramètres (réglage du filtre, simulation...).
2. Compléter le code avec les équations du filtre PF (prédiction, correction, ré-échantillonnage slide 22), le modèle dynamique et le modèle de mesure (slide 35) à l'aide de la fonction *hobs*($\mathbf{x}_k$, *params*) et commenter les résultats;
3. Faire varier le bruit de dynamique du filtre (matrice $\mathbf{Q}_f$) sur les différentes variables d'état et expliquer le comportement des particules.
4. Faire varier le bruit de mesure du filtre (matrice $\mathbf{R}_f$) entre 10^2 et 100^2 et expliquer les résultats.
5. Faire varier le nombre de particules N et expliquer les résultats.
6. Faire varier le seuil de ré-échantillonnage *threshold_resampling*. Pour les valeurs de 0, 0.5 et 1, tracer des histogrammes des poids (*hist(wp)*) et identifier le phénomène de dégénérescence et l'impact du ré-échantillonnage.
7. Simuler un trou de mesures entre $t = 50$ s et $t = 75$ s en utilisant la variable *is_measurementValid* et expliquer les résultats.
8. Modifier la fréquence des mesures (passer à 0.1 Hz) en utilisant la variable *is_measurementValid* et expliquer les résultats.
9. Proposer une autre façon de ré-échantillonner les poids, en remplacement de la fonction select.p (qui s'utilise comme: $indp = select(wp)$ avec wp l'ensemble des poids et $indp$ la liste des indices des nouvelles particules par rapport aux anciennes), la coder puis commenter les résultats. Vous pourrez vous aider de [12] (référence slide 36, pdf fourni avec les codes).

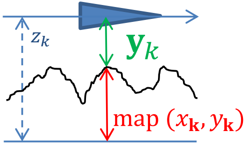
/!\ Les rapports de TP sont individuels.

State: $\mathbf{x}_k = [x_k, y_k, z_k, \theta_k]^T$, Control: $\mathbf{u}_k = [v_k, \omega_k]^T$

Dynamics:



$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) = \begin{bmatrix} x_k + v_k \Delta t \cos \theta_k \\ y_k + v_k \Delta t \sin \theta_k \\ z_k \\ \theta_k + \Delta t \omega_k \end{bmatrix}$$



Measurements (e.g., radar altimeter or laser telemeter):

$$\mathbf{y}_k = h(\mathbf{x}_k) + \mathbf{v}_k = z_k - \text{lectureCarte}(x_k, y_k) + \mathbf{v}_k$$

where $\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R})$ and $\mathbf{R} = 20^2\ \text{m}^2$.

Bibliography

1. Doucet, A., De Freitas, N., Gordon N. (2001). Sequential Monte Carlo Methods in Practice. Statistics for Engineering and Information Science. Springer, New York, NY.
2. Gordon, N. J., Salmond, D. J., Smith, A. F. (1993, April). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In IEE proceedings F (radar and signal processing) (Vol. 140, No. 2, pp. 107-113). IET Digital Library.
3. Burgers, G., Jan van Leeuwen, P., Evensen, G. (1998). Analysis scheme in the ensemble Kalman filter. Monthly weather review, 126(6), 1719-1724.
4. Oudjane, N., Musso, C. (2000, July). Progressive correction for regularized particle filters. In Proceedings of the Third International Conference on Information Fusion (Vol. 2, pp. THB2-10). IEEE.
5. Gordon, N., Ristic, B., Arulampalam, S. (2004). Beyond the kalman filter: Particle filters for tracking applications. Artech House, London, 830, 5.
6. Dahia, K. (2005). Nouvelles méthodes en filtrage particulaire-Application au recalage de navigation inertielle par mesures altimétriques (Doctoral dissertation, Université Joseph-Fourier-Grenoble I).
7. Särkkä, S., Vehtari, A., Lampinen, J. (2007). Rao-Blackwellized particle filter for multiple target tracking. Information Fusion, 8(1), 2-15.
8. Abdallah, F., Gning, A., Bonnifait, P. (2008). Box particle filtering for nonlinear state estimation using interval analysis. Automatica, 44(3), 807-815.
9. Merlinge, N. (2018). State estimation and trajectory planning using box particle kernels (Doctoral dissertation, Paris Saclay).
10. Palmier C., Dahia K., Merlinge N., Del Moral P., Laneuville D., Musso C. (2019). Adaptive Approximate Bayesian Computational Particle Filters for Underwater Terrain Aided Navigation. Information Fusion proc.
11. Fox, D. (2003). Adapting The Sample Size in Particle Filters Through KLD-Sampling, International Journal of Robotics Research, Vol. 22.
12. Li, T., Bolic, M., Djuric, P. M. (2015). Resampling methods for particle filtering: classification, implementation, and strategies. IEEE Signal processing magazine, 32(3), 70-86.