Intelligent distributed systems

Daniele Fontanelli

Department of Industrial Engineering
University of Trento
E-mail address: *daniele.fontanelli@unitn.it*

2022/2023

UNIVERSITÀ DI TRENTO | Dipartimento di Ingegneria Industriale

Outline

1. Unscented Kalman Filter
2. Particle Filters
3. Interactive Multiple Models
   - Generalised Pseudo-Bayesian Estimator
   - IMM

Outline

Unscented Kalman filter

The EKF relies on linearisation to propagate the mean and covariance of the state, which may lead to *overoptimistic estimates*, or even *inconsistency*, when the nonlinearities are severe.

Unscented Kalman filter

The EKF relies on linearisation to propagate the mean and covariance of the state, which may lead to *overoptimistic estimates*, or even *inconsistency*, when the nonlinearities are severe.
Another approach in this cases is the *Unscented Kalman Filter* (UKF), an extension of the Kalman filter that reduces the linearisation errors of the EKF. The use of the UKF can provide significant improvement over the EKF.

Unscented Kalman filter

Propagation of means

Consider a LIDAR acquiring data in *polar coordinates* $p = [r, \theta]^T$, i.e. r being the range and θ the angle.

Unscented Kalman filter

Propagation of means

Consider a LIDAR acquiring data in *polar coordinates* $p = [r, \theta]^T$, i.e. r being the range and θ the angle.
Assume that both the measurement quantities are affected by the noises η_r and η_θ that are generated by two *wide-sense stationary* and *independent processes* with symmetric pdfs and such that $\mu_{\eta_r} = \mu_{\eta_\theta} = 0$ and variances $\sigma^2_{\eta_r}$ and $\sigma^2_{\eta_\theta}$, i.e. $p_m = [r_m, \theta_m]^T = [r + \eta_r, \theta + \eta_\theta]^T$.

Unscented Kalman filter

Propagation of means

Consider a LIDAR acquiring data in *polar coordinates* $p = [r, \theta]^T$, i.e. r being the range and θ the angle.
Assume that both the measurement quantities are affected by the noises η_r and η_θ that are generated by two *wide-sense stationary* and *independent processes* with symmetric pdfs and such that $\mu_{\eta_r} = \mu_{\eta_\theta} = 0$ and variances $\sigma^2_{\eta_r}$ and $\sigma^2_{\eta_\theta}$, i.e. $p_m = [r_m, \theta_m]^T = [r + \eta_r, \theta + \eta_\theta]^T$.
We want to express those quantities in term of the vector of Cartesian coordinates $z = [x, y]^T$, therefore

$$z = \begin{bmatrix} r_m \cos(\theta_m) \\ r_m \sin(\theta_m) \end{bmatrix} = h(p_m)$$

Unscented Kalman filter

Propagation of means

Let us compute a first order approximation:

$$\mu_z = \mathsf{E}\left\{h(p_m)\right\} \approx \mathsf{E}\left\{h(\mathsf{E}\left\{p_m\right\}) + \left.\frac{\partial h(p_m)}{\partial p_m}\right|_{\mathsf{E}\{p_m\}} (p_m - \mathsf{E}\left\{p_m\right\})\right\}$$

Unscented Kalman filter

Propagation of means

Let us compute a first order approximation:

$$\mu_z = \mathsf{E}\left\{h(p_m)\right\} \approx \mathsf{E}\left\{h(\mathsf{E}\left\{p_m\right\}) + \left.\frac{\partial h(p_m)}{\partial p_m}\right|_{\mathsf{E}\{p_m\}} (p_m - \mathsf{E}\left\{p_m\right\})\right\}$$

Since $\mathsf{E}\left\{p_m\right\} = p$, we have

$$\mu_z \approx h(p) + \left.\frac{\partial h(p_m)}{\partial p_m}\right|_{p} \mathsf{E}\left\{p_m - p\right\} = h(p)$$

Unscented Kalman filter

Propagation of means

Let us compute a first order approximation:

$$\mu_z = \mathsf{E}\{h(p_m)\} \approx \mathsf{E}\left\{ h(\mathsf{E}\{p_m\}) + \left.\frac{\partial h(p_m)}{\partial p_m}\right|_{\mathsf{E}\{p_m\}} (p_m - \mathsf{E}\{p_m\}) \right\}$$

Since $\mathsf{E}\{p_m\} = p$, we have

$$\mu_z \approx h(p) + \left.\frac{\partial h(p_m)}{\partial p_m}\right|_{p} \mathsf{E}\{p_m - p\} = h(p)$$

Suppose $p = [1, \pi/2]^T$, we have $\mu_z = [0, 1]$.

Unscented Kalman filter

Propagation of means

However, if we consider

$$\mu_x = \mathsf{E}\{x_m\} = \mathsf{E}\{r_m \cos\theta_m\} = \mathsf{E}\{(r+\eta_r)(\cos\theta\cos\eta_\theta - \sin\theta\sin\eta_\theta)\}$$

that, assuming independence of the noises and symmetry of the pdfs and the point $p = [r, \theta]^T = [1, \pi/2]^T$, we have

$$\mu_x = r\cos\theta\mathsf{E}\{\cos\eta_\theta\} = 0$$

Unscented Kalman filter

Propagation of means

However, if we consider

$$\mu_x = \mathsf{E}\{x_m\} = \mathsf{E}\{r_m \cos\theta_m\} = \mathsf{E}\{(r+\eta_r)(\cos\theta\cos\eta_\theta - \sin\theta\sin\eta_\theta)\}$$

that, assuming independence of the noises and symmetry of the pdfs and the point $p = [r, \theta]^T = [1, \pi/2]^T$, we have

$$\mu_x = r\cos\theta\mathsf{E}\{\cos\eta_\theta\} = 0$$

If we consider

$$\mu_y = \mathsf{E}\{y_m\} = \mathsf{E}\{r_m \sin\theta_m\} = \mathsf{E}\{(r+\eta_r)(\sin\theta\cos\eta_\theta - \cos\theta\sin\eta_\theta)\}$$

that assuming independence of the noises and symmetry of the pdfs and the point $p = [r, \theta]^T = [1, \pi/2]^T$, we have

$$\mu_y = r\sin\theta\mathsf{E}\{\cos\eta_\theta\} = \mathsf{E}\{\cos\eta_\theta\}$$

Unscented Kalman filter

Propagation of means

Let us assume that η_θ is *uniformly* distributed (if no assumption is made, *it is not possible go any further*), with range in $\pm\eta_\theta^*$.

Unscented Kalman filter

Propagation of means

Let us assume that η_θ is *uniformly* distributed (if no assumption is made, *it is not possible go any further*), with range in $\pm\eta_\theta^*$.
Therefore, the *analytic solution* is

$$\mu_y = \mathsf{E}\{y_m\} = \mathsf{E}\{\cos\eta_\theta\} = \int_{-\eta_\theta^*}^{\eta_\theta^*} \cos\eta_\theta \frac{1}{2\eta_\theta^*} d\eta_\theta = \frac{\sin\eta_\theta^*}{\eta_\theta^*}$$

Unscented Kalman filter

Propagation of means

Let us assume that η_θ is *uniformly* distributed (if no assumption is made, *it is not possible go any further*), with range in $\pm\eta_\theta^*$.
Therefore, the *analytic solution* is

$$\mu_y = \mathsf{E}\{y_m\} = \mathsf{E}\{\cos\eta_\theta\} = \int_{-\eta_\theta^*}^{\eta_\theta^*} \cos\eta_\theta \frac{1}{2\eta_\theta^*} d\eta_\theta = \frac{\sin\eta_\theta^*}{\eta_\theta^*}$$

Notice that, while with the *first order approximation* we had $\mu_z = [0, 1]$, now instead of 1 we have a number that is equal to 1 *only if* $\eta_\theta^* = 0$, while a value that is < 1, $\forall \eta_\theta^* > 0$.

Unscented Kalman filter

Propagation of means

Let us assume that η_θ is *uniformly* distributed (if no assumption is made, *it is not possible go any further*), with range in $\pm\eta_\theta^*$.
Therefore, the *analytic solution* is

$$\mu_y = \mathsf{E}\{y_m\} = \mathsf{E}\{\cos\eta_\theta\} = \int_{-\eta_\theta^*}^{\eta_\theta^*} \cos\eta_\theta \frac{1}{2\eta_\theta^*} d\eta_\theta = \frac{\sin\eta_\theta^*}{\eta_\theta^*}$$

Notice that, while with the *first order approximation* we had $\mu_z = [0, 1]$, now instead of 1 we have a number that is equal to 1 *only if* $\eta_\theta^* = 0$, while a value that is < 1, $\forall \eta_\theta^* > 0$.
Hence, the *first order approximation* $\mu_z = [0, 1]$ is *incorrect*!

Unscented Kalman filter

Propagation of means

Let us see what happens if we expand up to the *second order*:

$$\mu_z = \mathsf{E}\left\{h(p_m)\right\} \approx \mathsf{E}\left\{h(\mathsf{E}\left\{p_m\right\}) + \left.\frac{\partial h(p_m)}{\partial p_m}\right|_{\mathsf{E}\{p_m\}} (p_m - \mathsf{E}\left\{p_m\right\})\right\} +$$
$$+ \frac{1}{2}\mathsf{E}\left\{\left.\frac{\partial^2 h(p_m)}{\partial r_m^2}\right|_{\mathsf{E}\{p_m\}} (r_m - \mathsf{E}\left\{r_m\right\})^2\right\} +$$
$$+ \frac{1}{2}\mathsf{E}\left\{\left.\frac{\partial^2 h(p_m)}{\partial \theta_m^2}\right|_{\mathsf{E}\{p_m\}} (\theta_m - \mathsf{E}\left\{\theta_m\right\})^2\right\} +$$
$$+ \mathsf{E}\left\{\left.\frac{\partial^2 h(p_m)}{\partial r_m \partial \theta_m}\right|_{\mathsf{E}\{p_m\}} (r_m - \mathsf{E}\left\{r_m\right\})(\theta_m - \mathsf{E}\left\{\theta_m\right\})\right\}$$

Unscented Kalman filter

Propagation of means

We notice that:

$$\frac{1}{2}\mathsf{E}\left\{ \left.\frac{\partial^2 h(p_m)}{\partial r_m^2}\right|_{\mathsf{E}\{p_m\}} (r_m - \mathsf{E}\{r_m\})^2 \right\} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Unscented Kalman filter

Propagation of means

We notice that:

$$\frac{1}{2}\mathsf{E}\left\{\left.\frac{\partial^2 h(p_m)}{\partial r_m^2}\right|_{\mathsf{E}\{p_m\}}(r_m-\mathsf{E}\{r_m\})^2\right\}=\begin{bmatrix}0\\0\end{bmatrix}$$

Then:

$$\mathsf{E}\left\{\left.\frac{\partial^2 h(p_m)}{\partial r_m\partial\theta_m}\right|_{\mathsf{E}\{p_m\}}(r_m-\mathsf{E}\{r_m\})(\theta_m-\mathsf{E}\{\theta_m\})\right\}=\begin{bmatrix}0\\0\end{bmatrix}$$

since the two **rv**s are *uncorrelated*.

Unscented Kalman filter

Propagation of means

We notice that:

$$\frac{1}{2}\mathsf{E}\left\{\left.\frac{\partial^2 h(p_m)}{\partial r_m^2}\right|_{\mathsf{E}\{p_m\}}(r_m-\mathsf{E}\{r_m\})^2\right\}=\begin{bmatrix}0\\0\end{bmatrix}$$

Then:

$$\mathsf{E}\left\{\left.\frac{\partial^2 h(p_m)}{\partial r_m\partial\theta_m}\right|_{\mathsf{E}\{p_m\}}(r_m-\mathsf{E}\{r_m\})(\theta_m-\mathsf{E}\{\theta_m\})\right\}=\begin{bmatrix}0\\0\end{bmatrix}$$

since the two **rv**s are *uncorrelated*.
Finally:

$$\frac{1}{2}\mathsf{E}\left\{\left.\frac{\partial^2 h(p_m)}{\partial\theta_m^2}\right|_{\mathsf{E}\{p_m\}}(\theta_m-\mathsf{E}\{\theta_m\})^2\right\}=\frac{1}{2}\sigma_{\eta_\theta}^2\begin{bmatrix}0\\-1\end{bmatrix}$$

Unscented Kalman filter

Propagation of means

From this result it follows that the *second order approximation*

$$\mu_z = \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \frac{1}{2}\sigma^2_{\eta_\theta} \begin{bmatrix} 0 \\ -1 \end{bmatrix}$$

Unscented Kalman filter

Propagation of means

From this result it follows that the *second order approximation*

$$\mu_z = \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \frac{1}{2}\sigma_{\eta_\theta}^2 \begin{bmatrix} 0 \\ -1 \end{bmatrix}$$

Hence, while $\mu_x = \mathsf{E}\{x_m\} = 0$, we have

$$\mu_y = \mathsf{E}\{y_m\} = \mathsf{E}\{\cos \eta_\theta\} \approx 1 - \frac{\sigma_{\eta_\theta}^2}{2}$$

where the approximation is *up to the second order*.

Unscented Kalman filter

Propagation of means

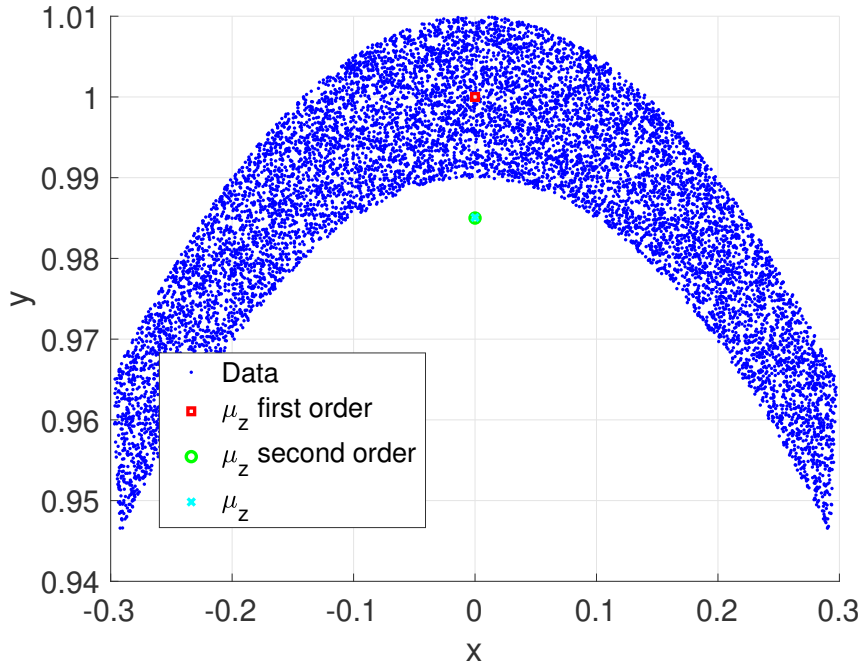

Figure: Approximation of the means for the previous example (with $\eta_r \sim \mathcal{U}(-0.01, 0.01)$ metres and $\eta_\theta \sim \mathcal{U}(-0.4, 0.4)$ radians) for the *first order* Taylor, *second order* Taylor and the solution using a Monte Carlo approach (i.e. *Type B evaluation*).

Unscented Kalman filter

Propagation of covariances

Let us try to compute the covariance of z, i.e.

$$\mathsf{C}\{z\} = \mathsf{E}\left\{(z-\mu_z)(z-\mu_z)^T\right\}$$

Unscented Kalman filter

Propagation of covariances

Let us try to compute the covariance of z, i.e.

$$\mathsf{C}\{z\} = \mathsf{E}\left\{(z-\mu_z)(z-\mu_z)^T\right\}$$

If we resort to the *first order approximation*, we have:

$$\mathsf{C}\{z\} \approx H\mathsf{C}\{p_m\}H^T$$

where

$$H = \left.\frac{\partial h(p_m)}{\partial p_m}\right|_{\mathsf{E}\{p_m\}} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

Unscented Kalman filter

Propagation of covariances

Notice that

$$\mathsf{C}\left\{p_m\right\} = \mathsf{E}\left\{(p_m - \mathsf{E}\left\{p_m\right\})(p_m - \mathsf{E}\left\{p_m\right\})^T\right\} =$$
$$= \mathsf{E}\left\{\begin{bmatrix}\eta_r \\ \eta_\theta\end{bmatrix}\begin{bmatrix}\eta_r & \eta_\theta\end{bmatrix}\right\} = \begin{bmatrix}\sigma_r^2 & 0 \\ 0 & \sigma_\theta^2\end{bmatrix}$$

Unscented Kalman filter

Propagation of covariances

Notice that

$$\mathsf{C}\left\{p_m\right\} = \mathsf{E}\left\{(p_m - \mathsf{E}\left\{p_m\right\})(p_m - \mathsf{E}\left\{p_m\right\})^T\right\} =$$
$$= \mathsf{E}\left\{\begin{bmatrix}\eta_r \\ \eta_\theta\end{bmatrix}\begin{bmatrix}\eta_r & \eta_\theta\end{bmatrix}\right\} = \begin{bmatrix}\sigma_r^2 & 0 \\ 0 & \sigma_\theta^2\end{bmatrix}$$

which leads to

$$\mathsf{C}\left\{z\right\} \approx H\mathsf{C}\left\{p_m\right\}H^T = \begin{bmatrix}\sigma_\theta^2 & 0 \\ 0 & \sigma_r^2\end{bmatrix}$$

i.e., an *approximated description* of the covariance.

Unscented Kalman filter

Propagation of covariances

Instead, if we make use of the approximation *up to the second order*, we have

$$\mathsf{C}\{z\} = \mathsf{E}\left\{(z-\mu_z)(z-\mu_z)^T\right\} =$$

$$= \mathsf{E}\left\{\begin{bmatrix} r_m\cos\theta_m \\ r_m\sin\theta_m - \left(1-\frac{\sigma_\theta^2}{2}\right)\end{bmatrix}\begin{bmatrix} r_m\cos\theta_m & r_m\sin\theta_m - \left(1-\frac{\sigma_\theta^2}{2}\right)\end{bmatrix}\right\} =$$

$$= \mathsf{E}\left\{\begin{bmatrix} r_m^2\cos\theta_m^2 & r_m^2\cos\theta_m\sin\theta_m - r_m\cos\theta_m\left(1-\frac{\sigma_\theta^2}{2}\right) \\ r_m^2\cos\theta_m\sin\theta_m - r_m\cos\theta_m\left(1-\frac{\sigma_\theta^2}{2}\right) & r_m^2\sin\theta_m^2 + \left(1-\frac{\sigma_\theta^2}{2}\right)^2 - 2r_m\sin\theta_m\left(1-\frac{\sigma_\theta^2}{2}\right)\end{bmatrix}\right\}$$

Unscented Kalman filter

Propagation of covariances

We now notice that if $\eta_r \sim \mathcal{U}(-r^*, r^*)$ with variance σ_r^2, $\eta_\theta \sim \mathcal{U}(-\eta_\theta^*, \eta_\theta^*)$ with variance σ_θ^2 and $p = [r, \theta]^T = [1, \pi/2]$, we have

$$\mathsf{E}\left\{r_m^2 \cos\theta_m^2\right\} = \mathsf{E}\left\{r_m^2\right\} \mathsf{E}\left\{\cos\theta_m^2\right\}$$

$$\mathsf{E}\left\{r_m^2\right\} = \mathsf{E}\left\{r^2\right\} + \mathsf{E}\left\{\eta_r^2\right\} + 2r\mathsf{E}\left\{\eta_r\right\} = 1 + \sigma_r^2$$

$$\mathsf{E}\left\{\cos\theta_m^2\right\} = \frac{1}{2}(1 + \mathsf{E}\left\{\cos 2\theta_m\right\})$$

$$\mathsf{E}\left\{\sin\theta_m^2\right\} = \frac{1}{2}(1 - \mathsf{E}\left\{\cos 2\theta_m\right\})$$

$$\mathsf{E}\left\{\cos 2\theta_m\right\} = -\mathsf{E}\left\{\cos 2\eta_\theta\right\}$$

$$\mathsf{E}\left\{\cos 2\eta_\theta\right\} \approx 1 - 2\sigma_\theta^2$$

$$\mathsf{E}\left\{\cos\theta_m\right\} = 0$$

$$\mathsf{E}\left\{\sin\theta_m\right\} = \mathsf{E}\left\{\cos\eta_\theta\right\} \approx 1 - \frac{\sigma_\theta^2}{2}$$

$$\mathsf{E}\left\{\cos\theta_m \sin\theta_m\right\} = -\sin_\theta^2 \mathsf{E}\left\{\cos\eta_\theta \sin\eta_\theta\right\} = 0$$

Unscented Kalman filter

Propagation of covariances

Substituting

$$\begin{aligned}\mathsf{C}\{z\} &= \mathsf{E}\left\{(z-\mu_z)(z-\mu_z)^T\right\} = \\ &= \begin{bmatrix}(1+\sigma_r^2)\sigma_\theta^2 & 0 \\ 0 & (1+\sigma_r^2)(1+\sigma_\theta^2) - \left(1-\frac{\sigma_\theta^2}{2}\right)^2\end{bmatrix}\end{aligned}$$

Unscented Kalman filter

Propagation of covariances

Instead, if we make use of the *analytic solution* with no approximation (which *is only valid* for uniform pdfs), we have

$$\mathsf{C}\{z\} = \mathsf{E}\left\{(z-\mu_z)(z-\mu_z)^T\right\} =$$

$$= \mathsf{E}\left\{\begin{bmatrix} r_m\cos\theta_m \\ r_m\sin\theta_m - \frac{\sin\eta_\theta^*}{\eta_\theta^*}\end{bmatrix}\begin{bmatrix} r_m\cos\theta_m & r_m\sin\theta_m - \frac{\sin\eta_\theta^*}{\eta_\theta^*}\end{bmatrix}\right\} =$$

$$= \mathsf{E}\left\{\begin{bmatrix} r_m^2\cos\theta_m^2 & r_m^2\cos\theta_m\sin\theta_m - r_m\cos\theta_m\frac{\sin}{\eta} \\ r_m^2\cos\theta_m\sin\theta_m - r_m\cos\theta_m\frac{\sin\eta_\theta^*}{\eta_\theta^*} & r_m^2\sin\theta_m^2 + \frac{\sin\eta_\theta^{*2}}{\eta_\theta^{*2}} - 2r_m\sin\theta_m\frac{s}{}\end{bmatrix}\right.$$

Unscented Kalman filter

Propagation of covariances

With respect to the previous case we only have the following differences

$$\mathsf{E}\left\{\cos 2\eta_\theta\right\} = \frac{\sin 2\eta_\theta^*}{2\eta_\theta^*}$$

$$\mathsf{E}\left\{\sin\theta_m\right\} = \mathsf{E}\left\{\cos\eta_\theta\right\} = \frac{\sin\eta_\theta^*}{\eta_\theta^*}$$

Unscented Kalman filter

Propagation of covariances

Hence, substituting we have

$$\mathsf{C}\{z\} = \mathsf{E}\left\{(z-\mu_z)(z-\mu_z)^T\right\} =$$

$$= \begin{bmatrix} \frac{1}{2}(1+\sigma_r^2)(1-\frac{\sin 2\eta_\theta^*}{2\eta_\theta^*}) & 0 \\ 0 & \frac{1}{2}(1+\sigma_r^2)(1+\frac{\sin 2\eta_\theta^*}{2\eta_\theta^*}) - \frac{\sin {\eta_\theta^*}^2}{{\eta_\theta^*}^2} \end{bmatrix}$$

Unscented Kalman filter

Propagation of covariances

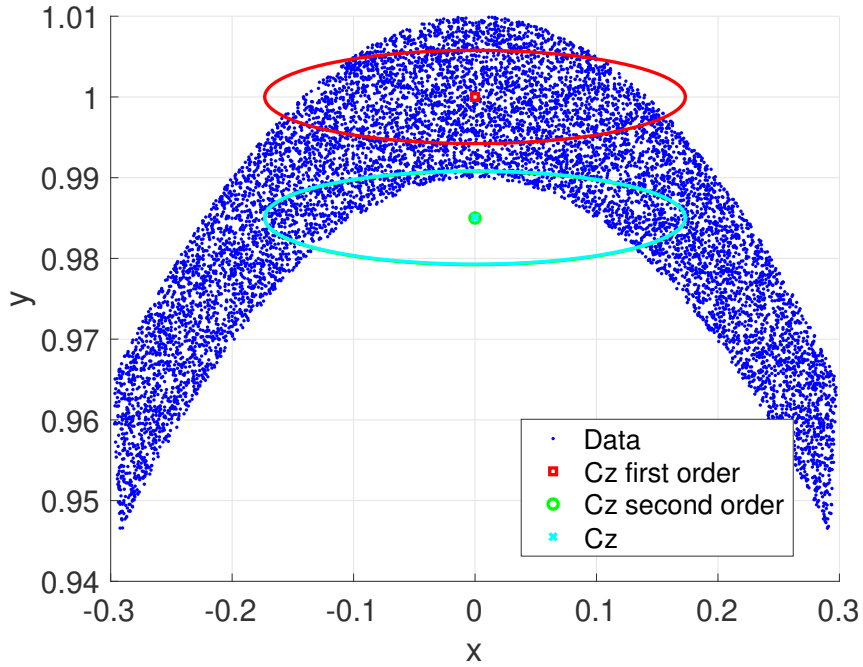

Figure: Approximation of the covariances for the previous example (with $\eta_r \sim \mathcal{U}(-0.01, 0.01)$ metres and $\eta_\theta \sim \mathcal{U}(-0.4, 0.4)$ radians) for the *first order* Taylor, *second order* Taylor and the *analytic* solution using a Monte Carlo approach (i.e. *Type B evaluation*).

Unscented Kalman filter

The main message is that a second order approximation *fits better* when the nonlinearities becomes relevant.

Unscented Kalman filter

The main message is that a second order approximation *fits better* when the nonlinearities becomes relevant.
The *mean* and the *variance* are not sufficient to fully describe the pdf in many cases.

Unscented Kalman filter

Propagation of covariances

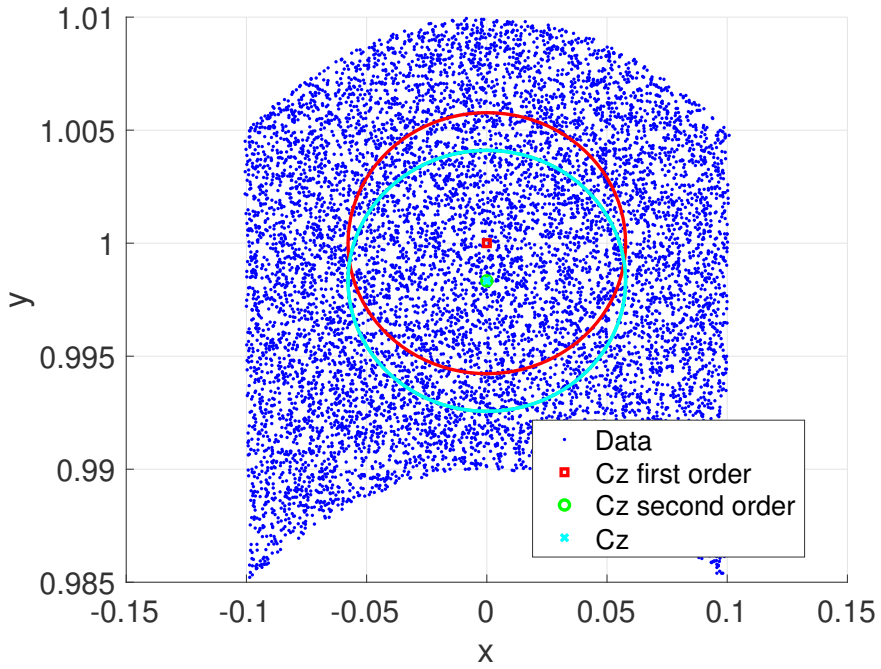

Figure: Approximation of the covariances for the previous example when $\eta_\theta \sim \mathcal{U}(-0.1, 0.1)$ radians.

Unscented Kalman filter

Unscented Transformation

It is in general quite difficult to determine the pdf $p(y)$ of y if $y = f(x)$. However:

- It is easy to compute $y^{(i)} = f(x^{(i)})$, where $x^{(i)}$ is a point extracted from $p(x)$;
- If we collect a set of $2n$ points $x^{(i)}$ (where n is the dimension of x), generating $y^{(i)} = f(x^{(i)})$, $i = 1, \ldots, 2n$, such that from $y^{(i)}$ it is possible to *approximate* $p(y)$, then the problem turns out to be feasible;
- Such points $x^{(i)}$ are called *sigma-points*.

Unscented Kalman filter

Unscented Transformation

More precisely:

- Suppose we know $\mathsf{E}\{x\} = \mu_x$ and $\mathsf{C}\{x\} = P_x$;
- We derive the $2n$ *sigma-points* $x^{(i)}$ such that the their *sample mean* and *sample covariance matrix* are μ_x and P_x;
- We then derive $y^{(i)} = f(x^{(i)})$, $i = 1, \ldots, 2n$;
- The *sample mean* $\hat{\mu}_y$ and *sample covariance matrix* $\hat{P}_y$ will then be, with a reasonable approximation, the actual μ_y and P_y of $p(y)$.

Unscented Kalman filter

Unscented Transformation

More precisely:

- Suppose we know $\mathsf{E}\{x\} = \mu_x$ and $\mathsf{C}\{x\} = P_x$;
- We derive the $2n$ *sigma-points* $x^{(i)}$ such that the their *sample mean* and *sample covariance matrix* are μ_x and P_x;
- We then derive $y^{(i)} = f(x^{(i)})$, $i = 1, \ldots, 2n$;
- The *sample mean* $\hat{\mu}_y$ and *sample covariance matrix* $\hat{P}_y$ will then be, with a reasonable approximation, the actual μ_y and P_y of $p(y)$.

The described algorithm is called the *unscented transformation*.

Unscented Kalman filter

Unscented Transformation

One way to generate the $2n$ *sigma-points* is the following:

- We denote with $\sqrt{nP_x}$ the square root of nP_x such that $\sqrt{nP_x}^T \sqrt{nP_x} = nP_x$ (this can be obtained with the *Cholesky decomposition*);
- We denote with $\left(\sqrt{nP_x}\right)_i$ the i-th row of $\sqrt{nP_x}$;
- The *sigma-points* are then possibly given by

$$
\begin{aligned}
x^{(i)} &= \mu_x + \tilde{x}^{(i)}, \quad i = 1, \dots, 2n \\
\tilde{x}^{(i)} &= \left(\sqrt{nP_x}\right)_i^T, \quad i = 1, \dots, n \\
\tilde{x}^{(i+n)} &= -\left(\sqrt{nP_x}\right)_i^T, \quad i = 1, \dots, n
\end{aligned}
$$

Unscented Kalman filter

Unscented Transformation

Given the *sigma-points*, is then possible to estimate the mean $\hat{\mu}_y$ and covariance matrix $\hat{P}_y$ of the actual μ_y and P_y as follows:

- $\hat{\mu}_y = \frac{1}{2n}\sum_{i=1}^{2n} y^{(i)} = \frac{1}{2n}\sum_{i=1}^{2n} f(x^{(i)})$;
- $\hat{P}_y = \frac{1}{2n}\sum_{i=1}^{2n}(y^{(i)} - \hat{\mu}_y)(y^{(i)} - \hat{\mu}_y)^T = \frac{1}{2n}\sum_{i=1}^{2n}(f(x^{(i)}) - \hat{\mu}_y)(f(x^{(i)}) - \hat{\mu}_y)^T$.

Unscented Kalman filter

Unscented Transformation

The main advantages of the *unscented transformation* are the following:

- There is no need to linearise the functions (as it happens, e.g., in the EKF);
- The approximation given by $\hat{\mu}_y$ is correct up to the *second order* of the Taylor expansion (not the first, as in the classic linearisation). To prove this, it is sufficient to compute the Taylor expansion of $\hat{\mu}_y$ around μ_x and then noticing that by the peculiar choice of the *sigma-points* (i.e., $\tilde{x}^{(i)} = -\tilde{x}^{(i+n)}$), the odd terms of the expansion cancels out;
- Instead, for the approximation given by $\hat{P}_y$, the order is the same as for the linearised version. Nonetheless, there is some level of knowledge of higher order terms (e.g., at least the sign is estimated correctly), hence a higher degree of approximation is obtained.

Unscented Kalman filter

The definition of the UKF then follows directly from the application of the *unscented transformation* to a Kalman Filter to propagate both the model transformation and the measurement transformation.

Unscented Kalman filter

The definition of the UKF then follows directly from the application of the *unscented transformation* to a Kalman Filter to propagate both the model transformation and the measurement transformation.
Therefore, given the following system definition

$$\begin{cases} x_{k+1} = f_k(x_k, u_k) + \nu_k \\ z_k = h_k(x_k) + \varepsilon_k \end{cases}$$

where ν_k and ε_k are two zero-mean uncertainties generated by a white stochastic process, with covariance matrices Q_k and R_k, respectively.

Unscented Kalman filter

The definition of the UKF then follows directly from the application of the *unscented transformation* to a Kalman Filter to propagate both the model transformation and the measurement transformation.
Therefore, given the following system definition

$$\begin{cases} x_{k+1} = f_k(x_k, u_k) + \nu_k \\ z_k = h_k(x_k) + \varepsilon_k \end{cases}$$

where ν_k and ε_k are two zero-mean uncertainties generated by a white stochastic process, with covariance matrices Q_k and R_k, respectively. Notice that the system can be time-varying, but the noises are at the moment considered as *additive* (e.g., a system that is affine in the *uncertainties*).

Unscented Kalman filter

Initialisation step:

- $$\hat{x}_0 = \mathsf{E}\{x_0\} \text{ and } P_0 = \mathsf{E}\left\{(x_0 - \hat{x}_0)(x_0 - \hat{x}_0)^T\right\}.$$

Unscented Kalman filter

Prediction step: Given $\hat{x}_{k-1}$ and P_{k-1}, compute

- *Sigma-points*:

$$\hat{x}_{k-1}^{(i)} = \hat{x}_{k-1} + \tilde{x}^{(i)}, \quad i = 1, \dots, 2n$$
$$\tilde{x}^{(i)} = \left(\sqrt{nP_{k-1}}\right)_i^T, \quad i = 1, \dots, n$$
$$\tilde{x}^{(i+n)} = -\left(\sqrt{nP_{k-1}}\right)_i^T, \quad i = 1, \dots, n$$

- Transform the *sigma-points* according to the system dynamics, i.e.

$$\hat{x}_k^{(i)} = f_k(\hat{x}_{k-1}^{(i)}, u_k).$$

Unscented Kalman filter

Prediction step:

- Compute the *predicted* estimates

$$\hat{x}_k^- = \frac{1}{2n} \sum_{i=1}^{2n} \hat{x}_k^{(i)}.$$

- Estimate the *predicted* covariance matrix of the estimation error

$$\hat{P}_k^- = \frac{1}{2n} \sum_{i=1}^{2n} (\hat{x}_k^{(i)} - \hat{x}_k^-)(\hat{x}_k^{(i)} - \hat{x}_k^-)^T + Q_{k-1}.$$

Unscented Kalman filter

Measurement update step:

- *Sigma-points*:

$$\hat{x}_k^{(i)} = \hat{x}_k^- + \tilde{x}^{(i)}, \quad i = 1, \dots, 2n$$
$$\tilde{x}^{(i)} = \left(\sqrt{nP_k^-}\right)_i^T, \quad i = 1, \dots, n$$
$$\tilde{x}^{(i+n)} = -\left(\sqrt{nP_k^-}\right)_i^T, \quad i = 1, \dots, n$$

  This step can use the same sigma-points of the prediction to save computation time, *at the price of a reduced performance*.

- Transform the *sigma-points* according to the system measurement function, i.e.

$$\hat{z}_k^{(i)} = h_k(\hat{x}_k^{(i)}).$$

Unscented Kalman filter

Measurement update step:

- Compute the *output* mean

$$\hat{z}_k = \frac{1}{2n} \sum_{i=1}^{2n} \hat{z}_k^{(i)}.$$

- Estimate the *innovation* covariance matrix

$$\hat{S}_z = \frac{1}{2n} \sum_{i=1}^{2n} (\hat{z}_k^{(i)} - \hat{z}_k)(\hat{z}_k^{(i)} - \hat{z}_k)^T + R_k.$$

- Estimate the *innovation* and state cross-covariance matrix

$$\hat{P}_{x,z} = \frac{1}{2n} \sum_{i=1}^{2n} (\hat{x}_k^{(i)} - \hat{x}_k^-)(\hat{z}_k^{(i)} - \hat{z}_k)^T.$$

Unscented Kalman filter

Measurement update step:

- Compute the UKF gain

$$W_k = \hat{P}_{x,z}\hat{S}_z^{-1}.$$

- Update the estimates

$$\hat{x}_k = \hat{x}_k^- + W_k(z_k - \hat{z}_k).$$

- Update the estimation error covariance matrix

$$P_k = P_k^- - W_k\hat{S}_z W_k^T.$$

- Then, starts over.

Unscented Kalman filter

Remark

If the uncertainties are not additive, i.e.

$$\begin{cases} x_{k+1} = f_k(x_k, u_k, \nu_k) \\ z_k = h_k(x_k, \varepsilon_k) \end{cases}$$

the problem can be solved by adding the uncertainties in the augmented state, i.e.

$$q_k = \begin{bmatrix} x_k \\ \nu_k \\ \varepsilon_k \end{bmatrix}.$$

Unscented Kalman filter

Remark

In such a case, we have

$$\hat{q}_0 = \begin{bmatrix} E\{x_0\} \\ 0 \\ 0 \end{bmatrix}$$

and

$$P_0 = \begin{bmatrix} E\{(x_0 - E\{x_0\})(x_0 - E\{x_0\})^T\} & 0 & 0 \\ 0 & Q_0 & 0 \\ 0 & 0 & R_0 \end{bmatrix}.$$

*Details can be found in S. Julier and J. Uhlmann, "*Unscented filtering and nonlinear estimation*", Proceedings of the IEEE, 92(3), pp. 401-422 (March 2004).*

Unscented Kalman filter

Remark

There are other possible unscented transformation that can be applied. The most popular are the generalised unscented transformation (using $2n+1$ points, an additional tuning parameter κ and generic weights of the sigma-points instead of $1/2n$), the simplex unscented transformation (which minimises the number of sigma-points to $n+1$) and the spherical unscented transformation (developed with the goal of rearranging the sigma-points of the simplex algorithm in order to obtain better numerical stability).

Unscented Kalman filter

Remark

UKF can be "easily" distributed among several robots using similar steps as for the EKF.
The interested reader may have a look at:
Wenling Li and Yingmin Jia, Consensus-Based Distributed Multiple Model UKF for Jump Markov Nonlinear Systems, *IEEE Transactions on Automatic Control, Vol. 57, No. 1, January 2012.*

Outline

Particle filter

Particle filters are a kind of *brute force* approaches developed in 40s, but actually applied from the 80s due to their relatively high computational burden.

Particle filter

Particle filters are a kind of *brute force* approaches developed in 40s, but actually applied from the 80s due to their relatively high computational burden.
The idea behind the *particle filter* is the same of the *frequentist* definition of probabilities: when a problem becomes too complicated, with many possible outcomes, it is impossible to give an analytic description, however an estimate of the probability may be given by *observation*.

Particle filter

Particle filters are a kind of *brute force* approaches developed in 40s, but actually applied from the 80s due to their relatively high computational burden.
The idea behind the *particle filter* is the same of the *frequentist* definition of probabilities: when a problem becomes too complicated, with many possible outcomes, it is impossible to give an analytic description, however an estimate of the probability may be given by *observation*.
We saw that UKF can be used when the EKF is a too harsh approximation. Nevertheless, both UKF and EKF need tuning. When the nonlinearities become too severe or the system is hard to be modelled, that is when the *Particle filter* comes into play.

Particle filter

We recall here the recursive Bayesian filter steps:

- The first computes the *new* estimate given all the previous measurements. In doing so, it makes use of the *system dynamics*, i.e.

$$p(x(k)|Z^{k-1}) = \int_{-\infty}^{+\infty} p(x(k)|x(k-1))p(x(k-1)|Z^{k-1})dx(k-1).$$

  This step is called *prediction*: it predicts the next value $x(k)$ given the previous knowledge $x(k-1)$;
- The second refines the prior with the *new set of measurements* (i.e. the new gained knowledge) using the likelihood function, i.e.

$$p(x(k)|Z^k) = \frac{p(z(k)|x(k))p(x(k)|Z^{k-1})}{p(z(k)|Z^{k-1})}.$$

  Since this second step updates the prediction with new knowledge, it is called *update*.

Particle filter

For the denominator of the Bayes equation (i.e., the normalisation factor), we consider the *Total Probability Law* and the Markovian property, thus having

$$p(z(k)|Z^{k-1}) = \int_{-\infty}^{+\infty} p(z(k)|x(k))p(x(k)|Z^{k-1})dx(k),$$

that is, as usual, the integral of the numerator (normalisation).

Particle filter

Linear system with Gaussian uncertainties

From the definition of the dynamic systems

$$x(k+1) = Ax(k) + \nu(k),$$

we have immediately that

$$\mathsf{V}\left\{x(k+1)\right\} = AP_kA^T + Q,$$

hence $\Sigma = Q$ i.e., the covariance matrix of the r.v. $x(k+1)$ given the model!

Particle filter

Linear system with Gaussian uncertainties

From the definition of the dynamic systems

$$x(k+1) = Ax(k) + \nu(k),$$

we have immediately that

$$\mathsf{V}\left\{x(k+1)\right\} = AP_kA^T + Q,$$

hence $\Sigma = Q$ i.e., the covariance matrix of the r.v. $x(k+1)$ given the model!
Notice how the uncertainty inevitably increases in the *prediction* step since AP_kA^T and Q are both p.d.

Particle filter

Linear system with Gaussian uncertainties

To summarise, assuming that $\mathsf{E}\{\nu(k)\nu(j)\} = Q(k)\delta_{k,j}$, $\nu(k) \sim \mathcal{N}(0, Q(k))$, $\mathsf{E}\{\varepsilon(k)\varepsilon(j)\} = R(k)\delta_{k,j}$, $\varepsilon(k) \sim \mathcal{N}(0, R(k))$, $\mathsf{E}\{\nu(k)\varepsilon(j)\} = 0$, $\forall i, j$, and

$$\begin{aligned} x(k+1) &= A(k)x(k) + B(k)u(k) + G(k)\nu(k), \\ z(k) &= H(k)x(k) + F(k)\varepsilon(k), \end{aligned}$$

we have

$$\begin{aligned} p(x(k)|Z^k) &= \mathcal{N}(\hat{x}(k), P(k)), \\ p(x(k+1)|x(k)) &= \mathcal{N}(A(k)\hat{x}(k) + B(k)u(k), G(k)Q(k)G(k)^T), \\ p(z(k+1)|x(k+1)) &= \mathcal{N}(H(k+1)\hat{x}(k+1), R(k+1)), \\ p(x(k+1)|Z^{k+1}) &= \mathcal{N}(\hat{x}(k+1), P(k+1)). \end{aligned}$$

Particle filter

In practice the *particle filter* represents a numerical implementation of a *Bayesian estimator*, considering a generic nonlinear system:

$$\begin{cases} x_{k+1} = f_k(x_k, u_k, \nu_k) \\ z_k = h_k(x_k, \varepsilon_k) \end{cases}$$

where ν_k and ε_k are assumed to be *independent* and *white*.

Particle filter

In practice the *particle filter* represents a numerical implementation of a *Bayesian estimator*, considering a generic nonlinear system:

$$\begin{cases} x_{k+1} = f_k(x_k, u_k, \nu_k) \\ z_k = h_k(x_k, \varepsilon_k) \end{cases}$$

where ν_k and ε_k are assumed to be *independent* and *white*.
The main objective of the *particle filter* is to approximate numerically $p(x(k+1)|Z^{k+1})$ using all the measurements up to time $k+1$ in a recursive way.

Particle filter

In practice the *particle filter* represents a numerical implementation of a *Bayesian estimator*, considering a generic nonlinear system:

$$\begin{cases} x_{k+1} = f_k(x_k, u_k, \nu_k) \\ z_k = h_k(x_k, \varepsilon_k) \end{cases}$$

where ν_k and ε_k are assumed to be *independent* and *white*.
The main objective of the *particle filter* is to approximate numerically $p(x(k+1)|Z^{k+1})$ using all the measurements up to time $k+1$ in a recursive way.
Notice that we assume that the first measurement is available at time $k = 1$, hence the filter is initialised trivially assuming $p(x_0) = p(x_0|Z^0)$, i.e., a prior.

Particle filter

Moreover, we assume that the pdfs involved in the *particle filter* estimator are generic.

Particle filter

Moreover, we assume that the pdfs involved in the *particle filter* estimator are generic.
Hence, they can be *multimodal*!

Particle filter

Moreover, we assume that the pdfs involved in the *particle filter* estimator are generic.
Hence, they can be *multimodal*!
Notice that in this case, assuming as the *best* estimate the *mean* can be a very poor choice (e.g., image a multimodal pdf symmetric with respect to the mean).

Particle filter

Initialisation step:

- At the beginning, a given number of n state vectors (i.e., *particles*) $x_0^{(i)}$, $i = 1, \ldots, n$, are randomly generated from $p(x_0)$, which is *assumed to be known*.

Particle filter

Prediction step:

- The *particles* are then propagated following the system dynamics, i.e.

$$x_{k+1}^{(i),-} = f_k(x_k^{(i)}, u_k, \nu_k),$$

where ν_k is *randomly generated* form the *known* pdf $p(\nu_k)$.

Particle filter

Measurement step: given $z_k = h_k(x_k, \varepsilon_k)$

- When the measurement z_{k+1} is available, the *conditional relative likelihood* is computed for each particle $x_{k+1}^{(i),-}$, that is we *numerically* evaluate $p(z_{k+1}|x_{k+1}^{(i),-})$.
- In practice, given the sensor readings $z^\star$, we compute numerically

$$\lambda_i = \Pr\left[z_{k+1} = z^\star | x_{k+1} = x_{k+1}^{(i),-}\right] \approx \Pr\left[\varepsilon_k = z^\star - h_k(x_{k+1}^{(i),-})\right],$$

that is the *probability that the i-th particle is correct given z_{k+1}*.

Particle filter

Measurement step:

- The probabilities are then normalised, i.e.

$$\lambda_i = \frac{\lambda_i}{\sum_{i=1}^{n} \lambda_i}.$$

- The particles are then *resampled* using λ_i. This resampling can be done in several ways. For example, for the i-th sample:
  - Generate a random number $y \sim \mathcal{U}(0,1)$. If $\sum_{i=1}^{j-1} \lambda_i < y$, but $\sum_{i=1}^{j} \lambda_i \geq y$, then $x_{k+1}^{(i)} = x_{k+1}^{(j),-}$ with probability π_j.
  - This resampling can be shown to converge towards the actual $p(x_{k+1}|Z^{k+1})$ when $n \to +\infty$.

Particle filter

Measurement step:

- After the resampling, it is then possible to have an estimate, e.g. the *conditional mean* $\mathrm{E}\left\{x_{k+1}|Z^{k+1}\right\}$, with the caveat mentioned previously for multimodal pdfs.

Particle filter

Example (Particle relative likelihood)

Consider $z_{k+1} = h_k(x_k) + \varepsilon_k$, where $z_{k+1} \in \mathbb{R}^m$, $\varepsilon_k \sim \mathcal{N}(0, R_k)$ and $z^\star$ is the actual sensor reading.

Particle filter

Example (Particle relative likelihood)

Consider $z_{k+1} = h_k(x_k) + \varepsilon_k$, where $z_{k+1} \in \mathbb{R}^m$, $\varepsilon_k \sim \mathcal{N}(0, R_k)$ and $z^\star$ is the actual sensor reading.
We have immediately that:

$$\lambda_i \propto \frac{1}{\sqrt{|2\pi R_k|}} e^{-\frac{1}{2}(z^\star - h(x_{k+1}^{(j),-}))^T R_k^{-1} (z^\star - h(x_{k+1}^{(j),-}))}.$$

Particle filter

Example (Particle relative likelihood)

Consider $z_{k+1} = h_k(x_k) + \varepsilon_k$, where $z_{k+1} \in \mathbb{R}^m$, $\varepsilon_k \sim \mathcal{N}(0, R_k)$ and $z^\star$ is the actual sensor reading.
We have immediately that:

$$\lambda_i \propto \frac{1}{\sqrt{|2\pi R_k|}} e^{-\frac{1}{2}(z^\star - h(x_{k+1}^{(j),-}))^T R_k^{-1} (z^\star - h(x_{k+1}^{(j),-}))}.$$

Notice that on the righthand side we do not have a probability (we missed the integral), but it is proportional to the probability.

Particle filter

Example (Particle relative likelihood)

Consider $z_{k+1} = h_k(x_k) + \varepsilon_k$, where $z_{k+1} \in \mathbb{R}^m$, $\varepsilon_k \sim \mathcal{N}(0, R_k)$ and $z^\star$ is the actual sensor reading.
We have immediately that:

$$\lambda_i \propto \frac{1}{\sqrt{|2\pi R_k|}} e^{-\frac{1}{2}(z^\star - h(x_{k+1}^{(j),-}))^T R_k^{-1} (z^\star - h(x_{k+1}^{(j),-}))}.$$

Notice that on the righthand side we do not have a probability (we missed the integral), but it is proportional to the probability.
This is the reason why we are normalising the probabilities.

Particle filter

Some additional comments:

- The *particle filter* suffers of *sample impoverishment*.
  - This is induced by the fact that the predicted pdf $p(x_{k+1}^{(j),-}|Z^k)$ is resampled over the relative likelihood $p(z_{k+1}|x_{k+1}^{(i),-})$, so it is expected that a large faction of the a-priori particles will have low likelihood probability. This will happen sooner or later.
  - The problem becomes even harsher if there are *modelling errors* in the output models.
  - Solution: *roughening*, *prior editing*, *regularisation*, *Markov Chain Monte Carlo resampling*, *auxiliary particle filtering*.

Particle filter

- Some solutions exist combining particle filtering with EKF or UKF, i.e.:
  - Each particle is *updated at the measurement time* using the EKF or the UKF;
  - Resampling is then performed using the measurement;
  - This is like running multiple KF, EKF or UKF Kalman filters (one for each particle) and then adding a resampling step after each measurement.

Particle filter

Remark

Particle filtering goes by many names: sequential importance sampling, bootstrap filtering, condensation algorithm, interacting particle approximations, Monte Carlo filtering, sequential Monte Carlo filtering, survival of the fittest.

Remark

If the system is generic, EKF works fine, UKF even better and particle filters even better. The increasing performance is paid with an increasing computation time.
If the system is linear and Gaussian, KF is the best (EKF is the same of the KF), while UKF and particle filters works as the KF but with an increasing computational load.

Particle filter

Remark

Particle filtering can be hardly distributed on several robots, due to the large amount of computations, hence communications, needed for the executions on several robots.
Nonetheless, once the estimate is available in one robot, it is just a matter of computing an, e.g., WLS among the estimates.

Outline

Interactive Multiple Models

Suppose we have a linear Gauss-Markov system as

$$x_{k+1} = A_i x_k + B_i u_k + G_i \nu_k,$$
$$z_k = H x_k + F_i \varepsilon_k,$$

where ν_k and ε_k are zero-mean, independent random stochastic processes.

Interactive Multiple Models

Suppose we have a linear Gauss-Markov system as

$$x_{k+1} = A_i x_k + B_i u_k + G_i \nu_k,$$
$$z_k = H x_k + F_i \varepsilon_k,$$

where ν_k and ε_k are zero-mean, independent random stochastic processes. Unfortunately, we do not know what is the model followed by the system dynamic, which may vary in a set $i = 1, \ldots, n$.

Interactive Multiple Models

Suppose we have a linear Gauss-Markov system as

$$x_{k+1} = A_i x_k + B_i u_k + G_i \nu_k,$$
$$z_k = H x_k + F_i \varepsilon_k,$$

where ν_k and ε_k are zero-mean, independent random stochastic processes. Unfortunately, we do not know what is the model followed by the system dynamic, which may vary in a set $i = 1, \ldots, n$.
How can we come up with a good estimate of $\hat{x}_k$?

Interactive Multiple Models

Suppose we have a linear Gauss-Markov system as

$$x_{k+1} = A_i x_k + B_i u_k + G_i \nu_k,$$
$$z_k = H x_k + F_i \varepsilon_k,$$

where ν_k and ε_k are zero-mean, independent random stochastic processes.
Unfortunately, we do not know what is the model followed by the system dynamic, which may vary in a set $i = 1, \ldots, n$.
How can we come up with a good estimate of $\hat{x}_k$?
As we did for the particle filter resampling, we may use the *evidence from the data*: the *likelihood*!

Interactive Multiple Models

Let us recall the Bayes example: *is Joe sick*?

Interactive Multiple Models

Let us recall the Bayes example: *is Joe sick*?
Joe carries out a set of health tests and it turns out that he results positive to a test about a very rare and bad disease affecting 0.1% of the population.

Interactive Multiple Models

Let us recall the Bayes example: *is Joe sick*?
Joe carries out a set of health tests and it turns out that he results positive to a test about a very rare and bad disease affecting 0.1% of the population.
The test *correctly* identifies 99% of the population having the disease and only 1% *wrongly* identifies persons who don't have the disease.

Interactive Multiple Models

Let us recall the Bayes example: *is Joe sick*?
Joe carries out a set of health tests and it turns out that he results positive to a test about a very rare and bad disease affecting 0.1% of the population.
The test *correctly* identifies 99% of the population having the disease and only 1% *wrongly* identifies persons who don't have the disease.
So, what is the probability that Joe actually has the disease?

Interactive Multiple Models

Let us define the event "Joe has the disease" with J, and with T "The test is positive".

Interactive Multiple Models

Let us define the event "Joe has the disease" with J, and with T "The test is positive".
Plugging all the numbers, there are good news for Joe, since we have:

$$\Pr[J|T] = \frac{\Pr[T|J]\Pr[J]}{\Pr[T|J]\Pr[J] + \Pr\left[T|\overline{J}\right]\Pr\left[\overline{J}\right]} = 9\%,$$

which is quite below the 99%!

Interactive Multiple Models

Let us define the event "Joe has the disease" with J, and with T "The test is positive".
Plugging all the numbers, there are good news for Joe, since we have:

$$\Pr[J|T] = \frac{\Pr[T|J]\Pr[J]}{\Pr[T|J]\Pr[J] + \Pr\left[T|\overline{J}\right]\Pr\left[\overline{J}\right]} = 9\%,$$

which is quite below the 99%!
This is an example of a *generative classifier*!

Interactive Multiple Models

Indeed, we can generalise the previous example: assuming that x is an n-dimensional vector of *features* of generic type (i.e., the test is positive or not) and y a set of possible *classes* (i.e., Joe is sick or not), we may write:

$$\Pr[y = c|x] = \frac{\Pr[x|y = c] \Pr[y = c]}{\sum_{\bar{c} \in \mathcal{C}} \Pr[x|y = \bar{c}] \Pr[y = \bar{c}]},$$

where $\mathcal{C}$ is the set of the possible classes.

Interactive Multiple Models

Indeed, we can generalise the previous example: assuming that x is an n-dimensional vector of *features* of generic type (i.e., the test is positive or not) and y a set of possible *classes* (i.e., Joe is sick or not), we may write:

$$\Pr[y = c|x] = \frac{\Pr[x|y = c] \Pr[y = c]}{\sum_{\bar{c} \in \mathcal{C}} \Pr[x|y = \bar{c}] \Pr[y = \bar{c}]},$$

where $\mathcal{C}$ is the set of the possible classes.
So, a *generative classifier* specifies how to generate the data using the *likelihood* $p(x|y = c)$ and the *prior* $p(y = c)$.

Interactive Multiple Models

Indeed, we can generalise the previous example: assuming that x is an n-dimensional vector of *features* of generic type (i.e., the test is positive or not) and y a set of possible *classes* (i.e., Joe is sick or not), we may write:

$$\Pr[y = c|x] = \frac{\Pr[x|y = c] \Pr[y = c]}{\sum_{\bar{c} \in \mathcal{C}} \Pr[x|y = \bar{c}] \Pr[y = \bar{c}]},$$

where $\mathcal{C}$ is the set of the possible classes.
So, a *generative classifier* specifies how to generate the data using the *likelihood* $p(x|y = c)$ and the *prior* $p(y = c)$.
Alternatively, we could fit directly the *posterior* $p(y = c|x)$, thus adopting a *discriminative classifier*.

Interactive Multiple Models

Since, in our problem, we have a *discrete set of possible system dynamics*, hence a set of possible classes $\mathcal{C}$ (where $y \in \mathcal{C}$), we will focus on how to use the likelihood $p(x|y = c)$, expressing *which kind of sensor measurements are we expecting from each model*, to determine the class $c \in \mathcal{C}$.

Interactive Multiple Models

Since, in our problem, we have a *discrete set of possible system dynamics*, hence a set of possible classes $\mathcal{C}$ (where $y \in \mathcal{C}$), we will focus on how to use the likelihood $p(x|y = c)$, expressing *which kind of sensor measurements are we expecting from each model*, to determine the class $c \in \mathcal{C}$.
To do so we have to design a *learning* algorithm working just on *positive examples*: we may say only that this measurement is compatible with the i-th system, not by enumerating all the other measurements that are not compatible with the i-th system.

Interactive Multiple Models

Since, in our problem, we have a *discrete set of possible system dynamics*, hence a set of possible classes $\mathcal{C}$ (where $y \in \mathcal{C}$), we will focus on how to use the likelihood $p(x|y = c)$, expressing *which kind of sensor measurements are we expecting from each model*, to determine the class $c \in \mathcal{C}$.
To do so we have to design a *learning* algorithm working just on *positive examples*: we may say only that this measurement is compatible with the i-th system, not by enumerating all the other measurements that are not compatible with the i-th system.
In practice, this is the way you teach a child what is a *cat*: you would say "Look, that is cat", not "Look that is not a cat" when you have in front a bird!

Interactive Multiple Models

This is similar to *binary classification*.

Interactive Multiple Models

This is similar to *binary classification*.
Indeed, the problem turns to *learn* a function $f(x)$ such that $f(x) = 1$ iff x is an example of the class (or *concept* $c \in \mathcal{C}$), otherwise $f(x) = 0$.

Interactive Multiple Models

This is similar to *binary classification*.
Indeed, the problem turns to *learn* a function $f(x)$ such that $f(x) = 1$ iff x is an example of the class (or *concept* $c \in \mathcal{C}$), otherwise $f(x) = 0$.
To account for *uncertainties* in the data, we resort to the *probabilistic description* of the problem.

Interactive Multiple Models

Example (K. Murphy - A Probabilistic Approach to Machine Learning)

Let us consider the *number game* (Tenenbaum, J., *A Bayesian framework for concept learning*. Ph.D. thesis, MIT, 1999).

Interactive Multiple Models

Example (K. Murphy - A Probabilistic Approach to Machine Learning)

Let us consider the *number game* (Tenenbaum, J., *A Bayesian framework for concept learning*. Ph.D. thesis, MIT, 1999).
We define simple arithmetical classes $\mathcal{C}$, such as "prime number" or "a number between 20 and 43".

Interactive Multiple Models

Example (K. Murphy - A Probabilistic Approach to Machine Learning)

Let us consider the *number game* (Tenenbaum, J., *A Bayesian framework for concept learning*. Ph.D. thesis, MIT, 1999).
We define simple arithmetical classes $\mathcal{C}$, such as "prime number" or "a number between 20 and 43".
Then, a set of numbers from a class $c \in \mathcal{C}$ are defined, i.e., $D = \{x_1, \ldots, x_n\}$. We have to determine if *a new* number $\bar{x}$ belongs to c, i.e. to *classify* $\bar{x}$.

Interactive Multiple Models

Example (Continued)

Suppose, that all numbers are integers between 1 and 100.

Interactive Multiple Models

Example (Continued)

Suppose, that all numbers are integers between 1 and 100.
Now suppose that $D = \{16\}$. What other number belongs to the class?

Interactive Multiple Models

Example (Continued)

Suppose, that all numbers are integers between 1 and 100.
Now suppose that $D = \{16\}$. What other number belongs to the class?
Of course, with only one number in D, the guess will be highly imprecise. Even if you expect something that is *close* to "16", what does it mean *close*? "15" is close, but any number ending with "6" will be close or any number starting with "1". Nonetheless, "89" is at this point *less likely* to happen.

Interactive Multiple Models

Example (Continued)

Suppose, that all numbers are integers between 1 and 100.
Now suppose that $D = \{16\}$. What other number belongs to the class?
Of course, with only one number in D, the guess will be highly imprecise.
Even if you expect something that is *close* to "16", what does it mean *close*? "15" is close, but any number ending with "6" will be close or any number starting with "1". Nonetheless, "89" is at this point *less likely* to happen.
More precisely, $\forall \bar{x} \in [1, 100]$, we can define the pmf $p(\bar{x}|D)$, which is called the *posterior predictive distribution*.

Interactive Multiple Models

Example (Continued)

Suppose, that all numbers are integers between 1 and 100.
Now suppose that $D = \{16\}$. What other number belongs to the class?
Of course, with only one number in D, the guess will be highly imprecise.
Even if you expect something that is *close* to "16", what does it mean *close*? "15" is close, but any number ending with "6" will be close or any number starting with "1". Nonetheless, "89" is at this point *less likely* to happen.
More precisely, $\forall \bar{x} \in [1, 100]$, we can define the pmf $p(\bar{x}|D)$, which is called the *posterior predictive distribution*.
With only $D = \{16\}$, $p(\bar{x}|D)$ will be quite spread.

Interactive Multiple Models

Example (Continued)

Suppose, that all numbers are integers between 1 and 100.
Now suppose that $D = \{16\}$. What other number belongs to the class?
Of course, with only one number in D, the guess will be highly imprecise.
Even if you expect something that is *close* to "16", what does it mean *close*? "15" is close, but any number ending with "6" will be close or any number starting with "1". Nonetheless, "89" is at this point *less likely* to happen.
More precisely, $\forall \bar{x} \in [1, 100]$, we can define the pmf $p(\bar{x}|D)$, which is called the *posterior predictive distribution*.
With only $D = \{16\}$, $p(\bar{x}|D)$ will be quite spread.
What happens if $D = \{16, 8, 2, 64\}$? Now you may conclude that either the class is *powers of two* or *even numbers* are good.

Interactive Multiple Models

Example (Continued)

Both hypotheses are correct and define the *space of hypotheses* $\mathcal{H}$ that are consistent with D, which is technically called *version space*. How to decide?

Interactive Multiple Models

Example (Continued)

Both hypotheses are correct and define the *space of hypotheses* $\mathcal{H}$ that are consistent with D, which is technically called *version space*. How to decide?

Most probably, you have decided for *powers of two* because it seems to be more appropriate. There is actually a *Bayesian explanation* for this.

Interactive Multiple Models

Example (Continued)

Both hypotheses are correct and define the *space of hypotheses* $\mathcal{H}$ that are consistent with D, which is technically called *version space*. How to decide?

Most probably, you have decided for *powers of two* because it seems to be more appropriate. There is actually a *Bayesian explanation* for this.

Let us define $\mathcal{H} = \{h_2, h_e\}$. The key intuition is that *we want to avoid suspicious coincidences*. If the true concept was even numbers, how come we only saw numbers that happened to be powers of two?

Interactive Multiple Models

Example (Continued)

More precisely, let us assume that the elements in D are *sampled uniformly at random* from the class c. Tenenbaum calls this *the strong sampling assumption*.

Interactive Multiple Models

Example (Continued)

More precisely, let us assume that the elements in D are *sampled uniformly at random* from the class c. Tenenbaum calls this *the strong sampling assumption*.

We have immediately, due to the independency assumption of the sampling, that $\forall h \in \mathcal{H}$ and for m randomly selected numbers

$$p(D|h) = \left(\frac{1}{\#h}\right)^m,$$

where $\#h$ are the number of elements verifying h.

Interactive Multiple Models

Example (Continued)

This probability embodies the *size principle*, which means that *the model should favour the simplest (smallest) hypothesis consistent with the data*.

Interactive Multiple Models

Example (Continued)

This probability embodies the *size principle*, which means that *the model should favour the simplest (smallest) hypothesis consistent with the data*. This is a quite known principle named *Occam's razor*, who was an English monk and philosopher, 1288–1348.

Interactive Multiple Models

Example (Continued)

This probability embodies the *size principle*, which means that *the model should favour the simplest (smallest) hypothesis consistent with the data*.
This is a quite known principle named *Occam's razor*, who was an English monk and philosopher, 1288–1348.
So, let us consider $D = \{16\}$. We have $p(D|h_2) = \frac{1}{6}$, while $p(D|h_e) = \frac{1}{50}$, so h_2 has a higher *likelihood* that h_e.

Interactive Multiple Models

Example (Continued)

This probability embodies the *size principle*, which means that *the model should favour the simplest (smallest) hypothesis consistent with the data*.
This is a quite known principle named *Occam's razor*, who was an English monk and philosopher, 1288–1348.
So, let us consider $D = \{16\}$. We have $p(D|h_2) = \frac{1}{6}$, while $p(D|h_e) = \frac{1}{50}$, so h_2 has a higher *likelihood* that h_e.
This becomes more evident when $D = \{16, 8, 2, 64\}$, since $p(D|h_2) = \frac{1}{6^4}$, while $p(D|h_e) = \frac{1}{50^4}$. Hence, h_2 has a *likelihood ratio* of $5000 : 1$ with respect to h_e: this is exactly the explanation of the impression we had.

Interactive Multiple Models

Example (Continued)

In order to apply the *Bayes theorem*, we have to combine the *prior* and the *likelihood*.

Interactive Multiple Models

Example (Continued)

In order to apply the *Bayes theorem*, we have to combine the *prior* and the *likelihood*.

As we mentioned several times, the *prior* expresses the *subjective* idea we have about the problem. For example, we may say that $p(h_2) = p(h_e)$, while the probability of $h_2^\star = \{\text{powers of } 2 \text{ without } 32\}$ can be assumed to be quite less than that of h_2.

Interactive Multiple Models

Example (Continued)

In order to apply the *Bayes theorem*, we have to combine the *prior* and the *likelihood*.

As we mentioned several times, the *prior* expresses the *subjective* idea we have about the problem. For example, we may say that $p(h_2) = p(h_e)$, while the probability of $h_2^\star = \{\text{powers of } 2 \text{ without } 32\}$ can be assumed to be quite less than that of h_2.

We can now apply the *Bayes theorem*

$$p(h|D) = \frac{p(D|h)p(h)}{\sum_{\bar{h}\in\mathcal{H}} p(D|\bar{h})p(\bar{h})} = \frac{\frac{\mathbb{I}(D\in h)}{\#h^m}p(h)}{\sum_{\bar{h}\in\mathcal{H}} \frac{\mathbb{I}(D\in \bar{h})}{\#\bar{h}^m}p(\bar{h})}.$$

where $\mathbb{I}(D \in h)$ is an *indicator function*, i.e., a function returning 1 iff all the elements in D verifies h.

Interactive Multiple Models

Example (Continued)

Now we can compute the correct hypothesis using a MAP estimator, i.e.

$$h^\star = \arg\max_{h \in \mathcal{H}} p(D|h)p(h).$$

Interactive Multiple Models

Example (Continued)

Now we can compute the correct hypothesis using a MAP estimator, i.e.

$$h^\star = \arg\max_{h\in\mathcal{H}} p(D|h)p(h).$$

Notice that a ML estimator will be instead like

$$h^\star = \arg\max_{h\in\mathcal{H}} p(D|h).$$

Interactive Multiple Models

Example (Continued)

Now we can compute the correct hypothesis using a MAP estimator, i.e.

$$h^\star = \arg\max_{h\in\mathcal{H}} p(D|h)p(h).$$

Notice that a ML estimator will be instead like

$$h^\star = \arg\max_{h\in\mathcal{H}} p(D|h).$$

If we have enough data, we see that the *data overwhelms the prior* and MAP $\rightarrow$ ML.

Interactive Multiple Models

Example (Continued)

Now we can compute the correct hypothesis using a MAP estimator, i.e.

$$h^\star = \arg\max_{h \in \mathcal{H}} p(D|h)p(h).$$

Notice that a ML estimator will be instead like

$$h^\star = \arg\max_{h \in \mathcal{H}} p(D|h).$$

If we have enough data, we see that the *data overwhelms the prior* and MAP $\rightarrow$ ML.
Moreover, if the set of hypothesis $\mathcal{H}$, we will *converge* to the true hypothesis (i.e., the MAP and the MI are *consistent*) when *the amount of data tends towards infinity*.

Interactive Multiple Models

Example (Continued)

Now we can compute the correct hypothesis using a MAP estimator, i.e.

$$h^\star = \arg\max_{h \in \mathcal{H}} p(D|h)p(h).$$

Notice that a ML estimator will be instead like

$$h^\star = \arg\max_{h \in \mathcal{H}} p(D|h).$$

If we have enough data, we see that the *data overwhelms the prior* and MAP $\rightarrow$ ML.
Moreover, if the set of hypothesis $\mathcal{H}$, we will *converge* to the true hypothesis (i.e., the MAP and the MI are *consistent*) when *the amount of data tends towards infinity*.
This concept has been discovered several times with, e.g., Type A analysis, Law of Large Numbers, sample mean and covariance, etc.

Interactive Multiple Models

Example (Continued)

Lets go back to our *posterior predictive distribution* $p(\bar{x}|D)$. We simply adopt here the *scientific method*: we determine *if our hypothesis verifies the observations* and to what extent.

Interactive Multiple Models

Example (Continued)

Lets go back to our *posterior predictive distribution* $p(\bar{x}|D)$. We simply adopt here the *scientific method*: we determine *if our hypothesis verifies the observations* and to what extent.
This always happens when we have multiple hypothesis verifying the data.
In such a case we may use a *Bayes model averaging*

$$p(\bar{x} = c|D) = \sum_{h \in \mathcal{H}} p(y = c|\bar{x}, h)p(h|D).$$

Interactive Multiple Models

Let us go back to our linear Gauss-Markov system

$$x_{k+1} = A_i x_k + B_i u_k + G_i \nu_k,$$
$$z_k = H x_k + F_i \varepsilon_k,$$

where ν_k and ε_k are zero-mean, independent random stochastic processes.

Interactive Multiple Models

Let us go back to our linear Gauss-Markov system

$$x_{k+1} = A_i x_k + B_i u_k + G_i \nu_k,$$
$$z_k = H x_k + F_i \varepsilon_k,$$

where ν_k and ε_k are zero-mean, independent random stochastic processes. We identify each model with $M_i = \{A_i, B_i, G_i, F_i\}$, $i = 1, \ldots, n$.

Interactive Multiple Models

Let us go back to our linear Gauss-Markov system

$$x_{k+1} = A_i x_k + B_i u_k + G_i \nu_k,$$
$$z_k = H x_k + F_i \varepsilon_k,$$

where ν_k and ε_k are zero-mean, independent random stochastic processes.
We identify each model with $M_i = \{A_i, B_i, G_i, F_i\}$, $i = 1, \ldots, n$.
To identify the correct model, we use the *Bayesian approach* just discussed: we start with a *prior* identifying the correct model, and we determine the *posterior* given the *likelihood*.

Interactive Multiple Models

By denoting the *prior* information with Z^0, we have that

$$\Pr\left[M_i|Z^0\right] = \lambda_{0,i},$$

with

$$\sum_{i=1}^{n} \lambda_{0,i} = 1.$$

Interactive Multiple Models

Static Multiple Model

If the model *does not change in time*, i.e., *static multiple model*, it is possible to compute $\lambda_{k,i}$ noticing that

$$\lambda_{k,i} = \Pr\left[M_i | Z^k\right] = \Pr\left[M_i | z_k, Z^{k-1}\right]$$

and then applying the *Bayes theorem*

$$\lambda_{k,i} = \Pr\left[M_i | z_k, Z^{k-1}\right] = \frac{p(z_k | Z^{k-1}, M_i) \Pr\left[M_i | Z^{k-1}\right]}{p(z_k | Z^{k-1})}.$$

Interactive Multiple Models

Static Multiple Model

If the model *does not change in time*, i.e., *static multiple model*, it is possible to compute $\lambda_{k,i}$ noticing that

$$\lambda_{k,i} = \text{Pr}\left[M_i|Z^k\right] = \text{Pr}\left[M_i|z_k, Z^{k-1}\right]$$

and then applying the *Bayes theorem*

$$\lambda_{k,i} = \text{Pr}\left[M_i|z_k, Z^{k-1}\right] = \frac{p(z_k|Z^{k-1}, M_i)\text{Pr}\left[M_i|Z^{k-1}\right]}{p(z_k|Z^{k-1})}.$$

Finally, applying the *Total Probability Law* at the denominator and recalling the definition of $\lambda_{k,i}$, we have $foralli = 1, \ldots, n$

$$\lambda_{k,i} = \frac{p(z_k|Z^{k-1}, M_i)\lambda_{k-1,i}}{\sum_{j=1}^{n} p(z_k|Z^{k-1}, M_j)\lambda_{k-1,j}}$$

Interactive Multiple Models

Static Multiple Model

We then recognise that the *likelihood*

$$p(z_k|Z^{k-1}, M_i) = p(F_i\varepsilon_k) = p(\eta_{k,i}).$$

Interactive Multiple Models

Static Multiple Model

We then recognise that the *likelihood*

$$p(z_k|Z^{k-1}, M_i) = p(F_i\varepsilon_k) = p(\eta_{k,i}).$$

In general, we can assume

$$p(\eta_{k,i}) = \mathcal{N}(0, R_{k,i}),$$

which turns out to be an approximated solution in the general case.

Interactive Multiple Models

Static Multiple Model

We then recognise that the *likelihood*

$$p(z_k|Z^{k-1}, M_i) = p(F_i\varepsilon_k) = p(\eta_{k,i}).$$

In general, we can assume

$$p(\eta_{k,i}) = \mathcal{N}(0, R_{k,i}),$$

which turns out to be an approximated solution in the general case. Moreover, we can have a value $\Lambda_{k,i}$ for the likelihood that is proportional to the actual value by computing, as done for the *particle filter*, i.e.

$$\Lambda_{k,i} \propto \frac{1}{\sqrt{|2\pi R_{k,i}|}} e^{-\frac{1}{2}(z_k - Hx_k^i)^T R_{k,i}^{-1}(z_k - Hx_k^i)}.$$

Interactive Multiple Models

Static Multiple Model

Hence, $\Lambda_{k,i}$ should be normalised as in the *particle filter* case.

Interactive Multiple Models

Static Multiple Model

Hence, $\Lambda_{k,i}$ should be normalised as in the *particle filter* case. However, recalling that

$$\lambda_{k,i} = \frac{p(z_k|Z^{k-1}, M_i)\lambda_{k-1,i}}{\sum_{j=1}^{n} p(z_k|Z^{k-1}, M_j)\lambda_{k-1,j}}$$

we have by design that

$$\sum_{i=1}^{n} \lambda_{k,i} = 1, \ \ \forall k > 0$$

Interactive Multiple Models

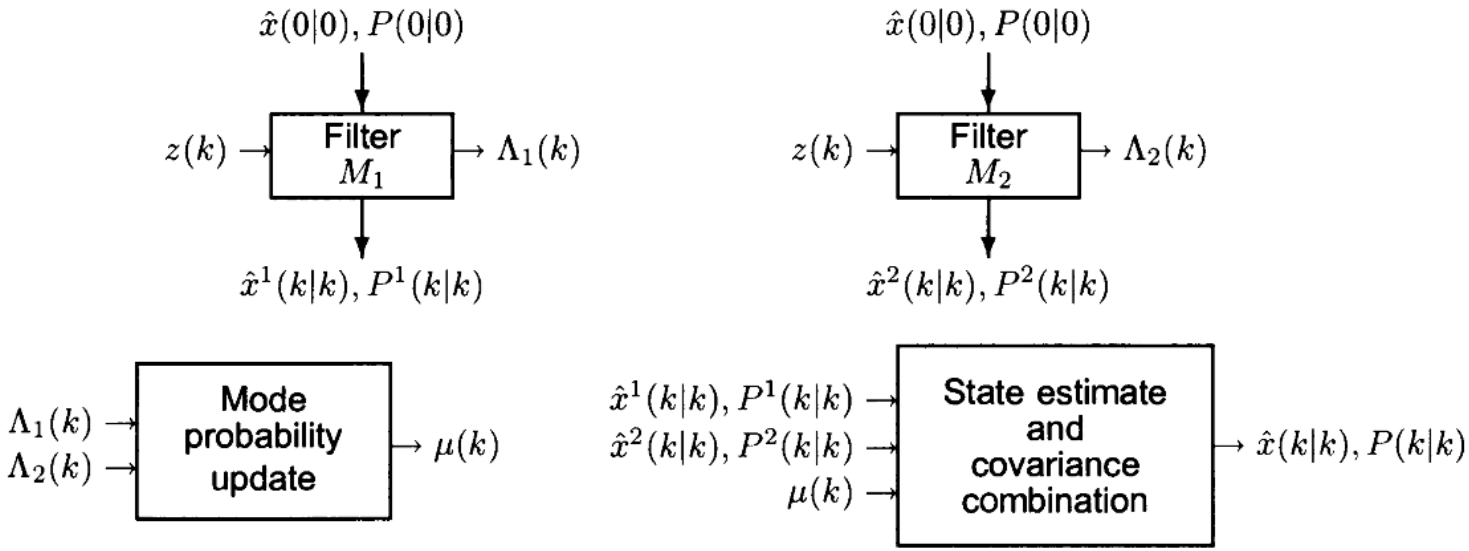

Figure: Static Multiple Models (courtesy of Yaakov Bar-Shalom, X. Rong Li, Thiagalingam Kirubarajan, *Estimation with Applications to Tracking and Navigation*, John Wiley Sons - 2001).

Interactive Multiple Models

Static Multiple Model

Notice that each system *runs independently to each other*, so we have n different estimates, namely $\hat{x}_k^i$, $i = 1, \ldots, n$, i.e., *mode-conditioned estimates*.

Interactive Multiple Models

Static Multiple Model

Notice that each system *runs independently to each other*, so we have n different estimates, namely $\hat{x}_k^i$, $i = 1, \ldots, n$, i.e., *mode-conditioned estimates*.

If we are interested in one single estimate, we may have the estimate at time k given by the *Bayes model averaging* presented previously, i.e.

$$\hat{x}_k = \sum_{i=1}^{n} \lambda_{k,i} \hat{x}_k^i.$$

Interactive Multiple Models

Static Multiple Model

Notice that each system *runs independently to each other*, so we have n different estimates, namely $\hat{x}_k^i$, $i = 1, \ldots, n$, i.e., *mode-conditioned estimates*.

If we are interested in one single estimate, we may have the estimate at time k given by the *Bayes model averaging* presented previously, i.e.

$$\hat{x}_k = \sum_{i=1}^{n} \lambda_{k,i} \hat{x}_k^i.$$

The corresponding covariance matrix of the estimation error is given by

$$P_k = \sum_{i=1}^{n} \lambda_{k,i} \left[P_k^i + (\hat{x}_k^i - \hat{x}_k)(\hat{x}_k^i - \hat{x}_k)^T \right],$$

which is similar to the *particle filter* sample covariance matrix estimation.

Interactive Multiple Models

Static Multiple Model

Remark

The filter can be either the KF, EKF or UKF.

Interactive Multiple Models

Static Multiple Model

Remark

The filter can be either the KF, EKF or UKF.

Remark

The model converges to the correct mode providing that: 1. the initial model is included in the n models; 2. the same model remains active for the entire estimation period.

Interactive Multiple Models

Static Multiple Model

Remark

The filter can be either the KF, EKF or UKF.

Remark

The model converges to the correct mode providing that: 1. the initial model is included in the n models; 2. the same model remains active for the entire estimation period.

Remark

If the initial model is not included in the n models, than the probabilities converge towards the nearest model.

Outline

Interactive Multiple Models

Generalised Pseudo-Bayesian

If the model is not static but switches among multiple modes, i.e., $M_{k,i}$, then previous approach cannot be used.

Interactive Multiple Models

Generalised Pseudo-Bayesian

If the model is not static but switches among multiple modes, i.e., $M_{k,i}$, then previous approach cannot be used.
We first assume that the models switchings is *Markovian*, i.e.

$$\Pr\left[M_{i,k}|M_{J,k-1}\right] = \Pr\left[M_{i,k}|M_{j,k}\right] = p_{ij}$$

where J is the sequence of past models the system had up to time $k-1$.

Interactive Multiple Models

Generalised Pseudo-Bayesian

If the model is not static but switches among multiple modes, i.e., $M_{k,i}$, then previous approach cannot be used.
We first assume that the models switchings is *Markovian*, i.e.

$$\Pr\left[M_{i,k}|M_{J,k-1}\right] = \Pr\left[M_{i,k}|M_{j,k}\right] = p_{ij}$$

where J is the sequence of past models the system had up to time $k-1$. Notwithstanding the Markovian property that simplifies the computation, we can show that the computation of the mode probabilities $\lambda_{k,i}$ implies the knowledge of all the past histories. This leads to a *combinatorial explosion* of the problem.

Interactive Multiple Models

Generalised Pseudo-Bayesian

If the model is not static but switches among multiple modes, i.e., $M_{k,i}$, then previous approach cannot be used.
We first assume that the models switchings is *Markovian*, i.e.

$$\Pr\left[M_{i,k}|M_{J,k-1}\right] = \Pr\left[M_{i,k}|M_{j,k}\right] = p_{ij}$$

where J is the sequence of past models the system had up to time $k-1$.
Notwithstanding the Markovian property that simplifies the computation, we can show that the computation of the mode probabilities $\lambda_{k,i}$ implies the knowledge of all the past histories. This leads to a *combinatorial explosion* of the problem.
How to solve this problem?

Interactive Multiple Models

Generalised Pseudo-Bayesian

An approximated solution can be found using the *Generalised Pseudo-Bayesian*: in practice we *fuse the past history at each step*.

Interactive Multiple Models

Generalised Pseudo-Bayesian

An approximated solution can be found using the *Generalised Pseudo-Bayesian*: in practice we *fuse the past history at each step*.
In particular, with the *Generalised Pseudo-Bayesian of first order* (GPB1), we condensate the n mode estimates $\hat{x}_k^i$, $i = 1, \ldots, n$ into one single estimate $\hat{x}_k$ (and associated covariance matrix P_k).

Interactive Multiple Models

Generalised Pseudo-Bayesian

An approximated solution can be found using the *Generalised Pseudo-Bayesian*: in practice we *fuse the past history at each step*.
In particular, with the *Generalised Pseudo-Bayesian of first order* (GPB1), we condensate the n mode estimates $\hat{x}_k^i$, $i = 1, \ldots, n$ into one single estimate $\hat{x}_k$ (and associated covariance matrix P_k).
Then all the filters starts over from the same estimate.

Interactive Multiple Models

Generalised Pseudo-Bayesian

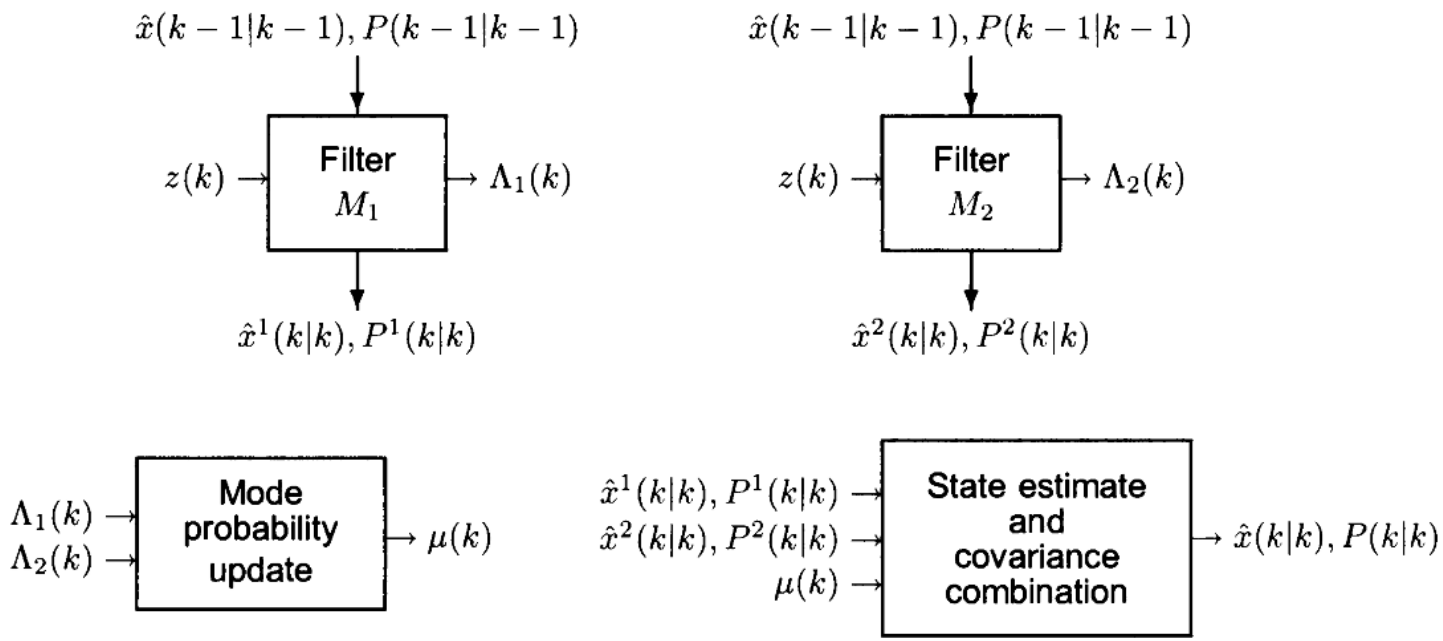

Figure: Gaussian Pseudo-Bayesian of first order (courtesy of Yaakov Bar-Shalom, X. Rong Li, Thiagalingam Kirubarajan, *Estimation with Applications to Tracking and Navigation*, John Wiley Sons - 2001).

Interactive Multiple Models

Generalised Pseudo-Bayesian: First order

To carry out the *history annihilation* we resort to this approximation:

$$
\begin{aligned}
p(x_k|Z^k) &= \sum_{i=1}^{n} p(x_k|M_{k,i}, Z^k)\mathrm{Pr}\left[M_{k,i}|Z^k\right] = \\
&= \sum_{i=1}^{n} p(x_k|M_{k,i}, z_k, Z^{k-1})\lambda_{k,i} \approx \\
&\approx \sum_{i=1}^{n} p(x_k|M_{k,i}, z_k, \hat{x}_{k-1}, P_{k-1})\lambda_{k,i}.
\end{aligned}
$$

Interactive Multiple Models

Generalised Pseudo-Bayesian: First order

Mode matching filter step:

- Given $\hat{x}_{k-1}$ and P_{k-1}, we first compute $\hat{x}_k^i$ and P_k^i, $\forall i = 1, \ldots, n$, using the usual KF, EKF or UKF steps;
- Similarly, we compute the likelihood as in the *static mode* case, i.e.

$$\Lambda_{k,i} = p(z_k | M_{k,i}, Z^{k-1}) = p(z_k | M_{k,i}, \hat{x}_{k-1}, P_{k-1}).$$

Interactive Multiple Models

Generalised Pseudo-Bayesian: First order

Mode probability update step:

- The mode probability is given by

$$\begin{aligned}\lambda_{k,i} &= \Pr\left[M_{k,i}|Z^k\right] = \Pr\left[M_{k,i}|z_k, Z^{k-1}\right] = \\ &= \frac{1}{a}p(z_k|M_{k,i}, Z^{k-1})\Pr\left[M_{k,i}|Z^{k-1}\right] = \\ &= \frac{1}{a}\Lambda_{k,i}\sum_{j=1}^{n}\Pr\left[M_{k,i}|M_{k-1,j}Z^{k-1}\right]\Pr\left[M_{k-1,j}|Z^{k-1}\right] = \\ &= \frac{1}{a}\Lambda_{k,i}\sum_{j=1}^{n}p_{ji}\lambda_{k-1,j}.\end{aligned}$$

where, a is a normalisation constant, that is obviously given by

$$a = \sum_{i=1}^{n}\Lambda_{k,i}\sum_{j=1}^{n}p_{ji}\lambda_{k-1,j}.$$

Interactive Multiple Models

Generalised Pseudo-Bayesian: First order

State and covariance estimation step:

- Again we make use of the *Bayes model averaging*, i.e.

$$\hat{x}_k = \sum_{i=1}^{n} \lambda_{k,i} \hat{x}_k^i,$$

$$P_k = \sum_{i=1}^{n} \lambda_{k,i} \left[P_k^i + (\hat{x}_k^i - \hat{x}_k)(\hat{x}_k^i - \hat{x}_k)^T \right].$$

Interactive Multiple Models

Generalised Pseudo-Bayesian

In the *Generalised Pseudo-Bayesian of second order* (GPB2), we maintain the n mode estimates $\hat{x}^i_{k-1}$, $i = 1, \ldots, n$.

Interactive Multiple Models

Generalised Pseudo-Bayesian

In the *Generalised Pseudo-Bayesian of second order* (GPB2), we maintain the n mode estimates $\hat{x}^i_{k-1}$, $i = 1, \ldots, n$.
Then all the filters starts over from *each* of the n estimates $\hat{x}^i_{k-1}$, thus obtaining $\hat{x}^{i,j}_k$, $i, j = 1, \ldots, n$. This way we have n^2 estimates and n^2 likelihood probabilities $\Lambda_{i,j,k}$.

Interactive Multiple Models

Generalised Pseudo-Bayesian

In the *Generalised Pseudo-Bayesian of second order* (GPB2), we maintain the n mode estimates $\hat{x}^i_{k-1}$, $i = 1, \ldots, n$.
Then all the filters starts over from *each* of the n estimates $\hat{x}^i_{k-1}$, thus obtaining $\hat{x}^{i,j}_k$, $i, j = 1, \ldots, n$. This way we have n^2 estimates and n^2 likelihood probabilities $\Lambda_{i,j,k}$.
We then condensate the estimates executed on the same j-th filter into one single estimate, to have back n estimates $\hat{x}^i_k$, $i = 1, \ldots, n$, for the next step.

Interactive Multiple Models

Generalised Pseudo-Bayesian

In the *Generalised Pseudo-Bayesian of second order* (GPB2), we maintain the n mode estimates $\hat{x}^i_{k-1}$, $i = 1, \ldots, n$.
Then all the filters starts over from *each* of the n estimates $\hat{x}^i_{k-1}$, thus obtaining $\hat{x}^{i,j}_k$, $i, j = 1, \ldots, n$. This way we have n^2 estimates and n^2 likelihood probabilities $\Lambda_{i,j,k}$.
We then condensate the estimates executed on the same j-th filter into one single estimate, to have back n estimates $\hat{x}^i_k$, $i = 1, \ldots, n$, for the next step.
At each time step we can still obtain a single estimate $\hat{x}_k$ (and associated covariance matrix P_k), as we did for the *static mode*, but this estimate is *not* used for the next step.

Interactive Multiple Models

Generalised Pseudo-Bayesian

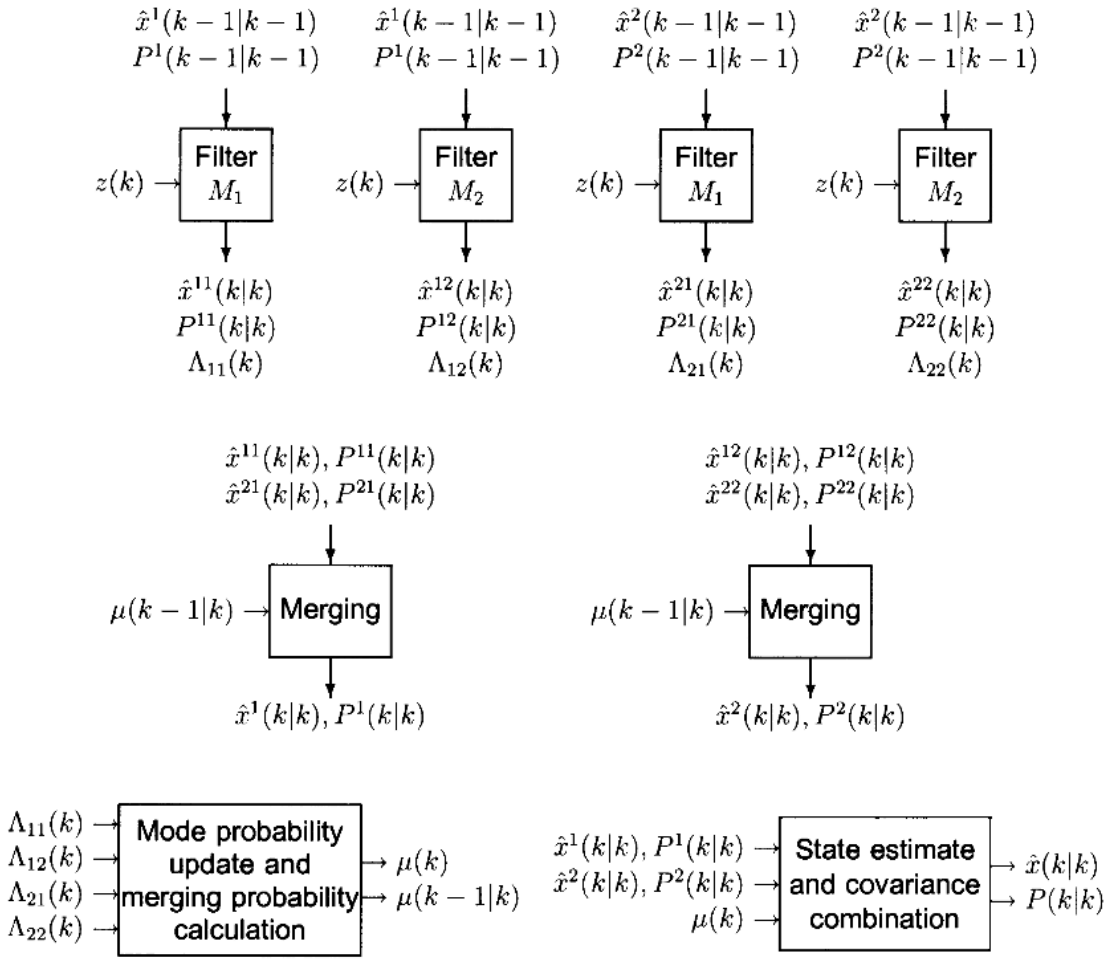

Figure: Gaussian Pseudo-Bayesian of second order (courtesy of Yaakov

Outline

Interactive Multiple Models

IMM

An additional solution is given by the *interactive multiple models* (IMM) filter.

Interactive Multiple Models

IMM

An additional solution is given by the *interactive multiple models* (IMM) filter.
The main difference is given by the estimates adopted by each filter at each iteration: it is not a unified estimate (as in GPB1), it is not the output of each filter (as in GPB2), but rather is a *mixture of the two*.

Interactive Multiple Models

IMM

An additional solution is given by the *interactive multiple models* (IMM) filter.
The main difference is given by the estimates adopted by each filter at each iteration: it is not a unified estimate (as in GPB1), it is not the output of each filter (as in GPB2), but rather is a *mixture of the two*.
The mixture is computed using the *mixing probabilities* $\lambda_{k,i}$, $i = 1, \ldots, n$.

Interactive Multiple Models

IMM

An additional solution is given by the *interactive multiple models* (IMM) filter.
The main difference is given by the estimates adopted by each filter at each iteration: it is not a unified estimate (as in GPB1), it is not the output of each filter (as in GPB2), but rather is a *mixture of the two*.
The mixture is computed using the *mixing probabilities* $\lambda_{k,i}$, $i = 1, \ldots, n$.
In other words, the n filters *interact* using the mixture of the estimates $\hat{x}_k^i$, $i = 1, \ldots, n$.

Interactive Multiple Models

IMM

An additional solution is given by the *interactive multiple models* (IMM) filter.
The main difference is given by the estimates adopted by each filter at each iteration: it is not a unified estimate (as in GPB1), it is not the output of each filter (as in GPB2), but rather is a *mixture of the two*.
The mixture is computed using the *mixing probabilities* $\lambda_{k,i}$, $i = 1, \ldots, n$.
In other words, the n filters *interact* using the mixture of the estimates $\hat{x}_k^i$, $i = 1, \ldots, n$.
The mixture assumes the form of a *Gaussian mixture model* that is then approximated via *moment matching* (i.e., extracting mean and covariance matrix) to a single Gaussian.

Interactive Multiple Models

IMM

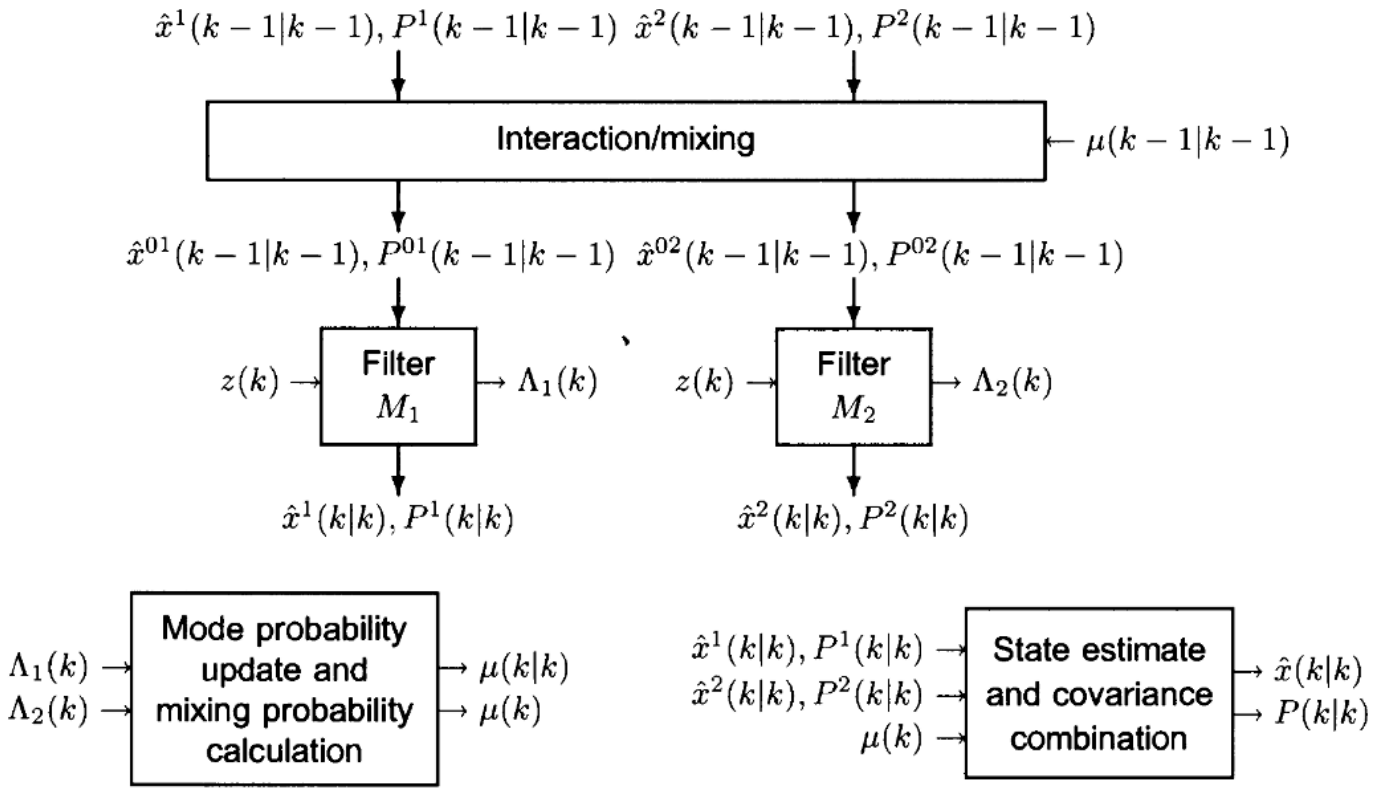

Figure: Interactive Multiple Models (courtesy of Yaakov Bar-Shalom, X. Rong Li, Thiagalingam Kirubarajan, *Estimation with Applications to Tracking and Navigation*, John Wiley Sons - 2001).

Interactive Multiple Models

IMM

Remark

The method can be applied to nonlinear systems using linearisation (i.e., EKF) or adopting more complicated schemes (i.e., UKF).

Interactive Multiple Models

IMM

Remark

The method can be applied to nonlinear systems using linearisation (i.e., EKF) or adopting more complicated schemes (i.e., UKF).

Remark

All the previous solutions can actually be applied to a distributed system suing linear consensus theory.