PERCEPTION – I

Estimation and Data Fusion

Robotics Winter School

Roland Chapuis

INSTITUT PASCAL
sciences de l'ingénierie et des systèmes

chapuis.roland@uca.fr

January 2019

Introduction

- Data Fusion aims at combining several sensors data to:
  - provide an accurate **estimation** of its state,
  - allow to take the best **decision**.
- We mainly concentrate here on the **estimation** and we'll address:
  - Bayesian estimation,
  - Kalman filtering,
  - Data fusion for estimation,
  - Fusion and Graphical Models.

Bayesian Estimation

Introduction

Robots usually embed many sensors.

- How can it make the best use of the data from these embedded sensors ?
- It depends on applications: pose estimations, control ?
- Here we mainly concentrate on the **data fusion** aiming at giving **an estimation** of an unknown vector state x thanks to the sensors data
- We look for a general framework not dedicated to a specific applications.
- The Bayesian Framework offers this possibility.

Bayesian estimation principles

- Bayesian estimation is the basis for parametric data fusion.
- We are looking for an estimation $\hat{x}$ of an unknown parameter x having a noisy measurement z that is linked to x by:

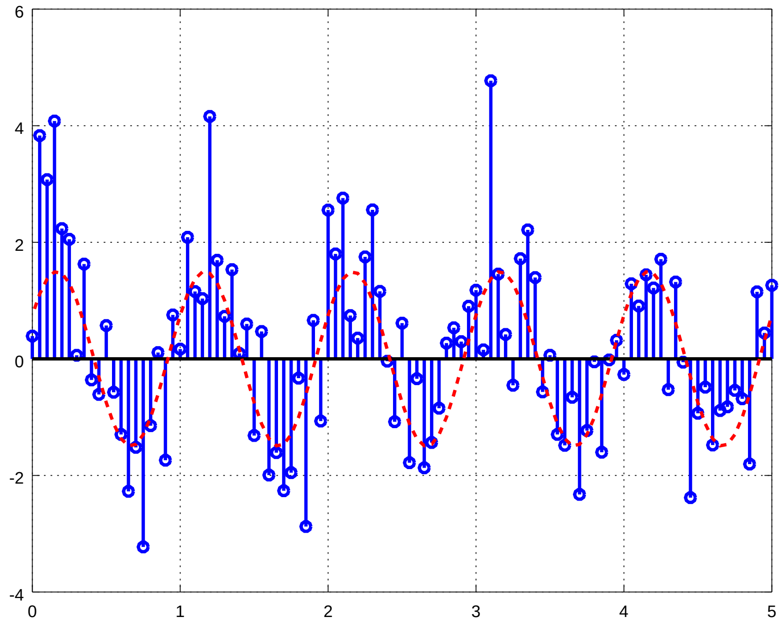
$$z = h(x, v) \quad (1)$$

  Notice x is usually a vector as well as z. However we keep this notation for sake of simplicity. v is the noise on z.

- We only concentrate (at the moment) on the estimation of x at a given time regardless the eventual dynamic evolution of the robot.

Example

We wish to estimate f_0, ϕ and A of a noised signal using the first z_n ($z \in [0, N[$) measurements: $z_n = A\sin(2\pi f_0 n + \phi) + v_n$ with $n \in [0, N[$



- We can write: $z = h(x, v)$

- With: $z = (z_0, z_1, \cdots, z_{N-1})^\top$, $x = (f_0, \phi, A)^\top$ and $v = (v_0, v_1, \cdots, v_{N-1})^\top$.

- We look for an **estimation** $\hat{x} = (\hat{f}_0, \hat{\phi}, \hat{A})^\top$ of $x = (f_0, \phi, A)^\top$ using $z = (z_0, z_1, \cdots, z_{N-1})^\top$.

- The estimation problem can be formulated as follows: **what is the best estimation $\hat{x}$ of $x = (f_0, \phi, A)^\top$ having z and knowing the statistical properties of the noise v** ?

- Because of this noise, we'll need to use probability tools to solve this problem.

Bayes rules

- We only consider the *continuous variables Bayes Rule* here. Consider x and y random values, we'll use their *pdf* [1]:
  - $p(x)$ and $p(y)$: *pdf* of x and y,
  - $p(x, y)$: joint *pdf* of x and y,
  - $p(x|y)$: conditional *pdf* of x having y
- We have:

$$p(x, y) = p(x|y)p(y) = p(y|x)p(x)$$

[1] Probability Density Function

and:

$$\boxed{p(x|y) = \frac{p(x,y)}{p(y)} = \frac{p(y|x)p(x)}{p(y)}} \tag{2}$$

- Since $p(y) = \int_{-\infty}^{+\infty} p(y|x)p(x)dx$, we'll have:

$$p(x|y) = \frac{p(y|x)p(x)}{\int_{-\infty}^{+\infty} p(y|x)p(x)dx}$$

- That can be generalized to multiple variables x_i for $i \in [1,n]$ and y_j for $j \in [1,m]$:

$$\boxed{p(x_1,\cdots,x_n|y_1,\cdots,y_m) = \frac{p(x_1,\cdots,x_n,y_1,\cdots,y_m)}{p(y_1,\cdots,y_m)} = \frac{p(y_1,\cdots y_m|x_1,\cdots,x_n)p(x_1,\cdots,x_n)}{p(y_1,\cdots,y_m)}}$$

Likelihood function

- Actually we are looking for x having z measurement.
- So we wish to know $p(x|z)$.
- Considering Bayes Equation, it is straightforward to obtain $p(x|z)$:

$$\boxed{p(x|z) = \frac{p(z|x)p(x)}{p(z)}}$$

- This is exactly what we want: getting x having z
- x and z are most of the time vectors
- However, we wish to get an **estimation** $\hat{x}$ of x (we'll see later),

$$p(x|z) = \frac{p(z|x)p(x)}{p(z)}$$

We have three parts: $p(z|x)$, $p(x)$ and $p(y)$.

- $p(z|x)$ is the link between the measurement z and the unknown x, this term is named **prior likelihood**,
- $p(x)$ is **prior knowledge** we have on x,
- $p(z)$ is the knowledge we have on z whatever x. Since $p(z)$ is no linked to x, $p(z)$ is rarely used in practice.

Likelihood function

$$p(x|z) = \frac{p(z|x)p(x)}{p(z)}$$

$p(z|x)$ can represent two things:

1. the *pdf* of measurement z knowing x.

   - Here z is a random variable while x is known, it is actually **the measurement characterization** knowing x.
   - $p(z|x)$ is typically given by eq. $z = h(x, v)$

2. $p(z|x)$ can also represent x while we have z:

- In this case $p(z|x)$ is the **likelihood function** of x. Here x is the random value and z is known.
- In order to mention explicitly that x is the random value, we write the likelihood function as:

$$p(x; z|x)$$

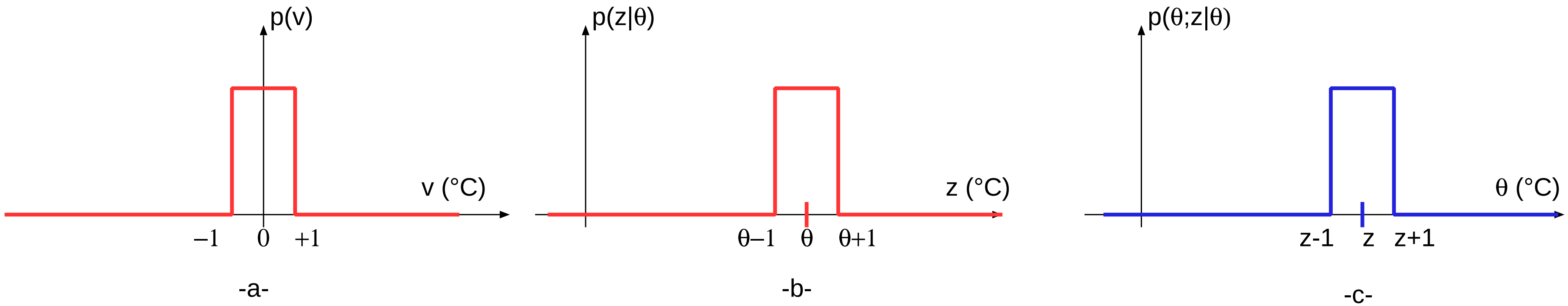
- Some authors [7] even write:

$$l(x; z) \triangleq p(x; z|x)$$

Example 1

Suppose the room temperature is $\theta = 20^\circ$. A sensor provides noised measurement z with a $\pm 1^\circ$ uniform error.

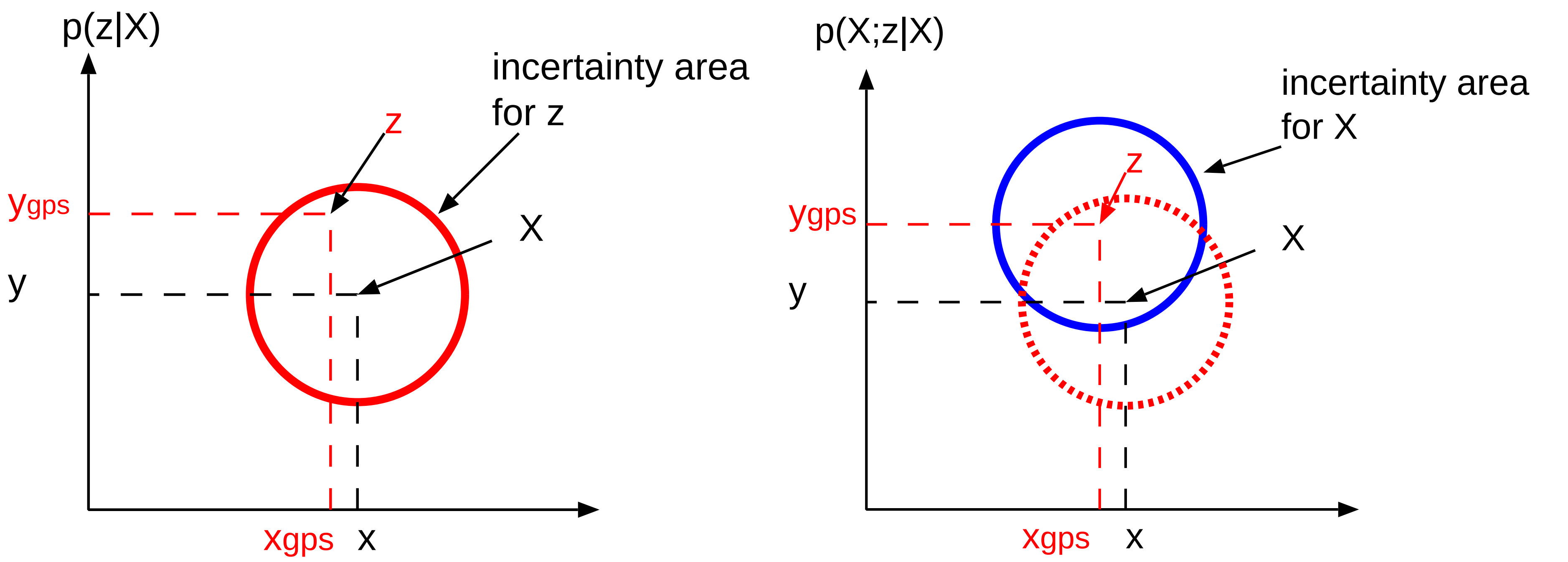
- We can write: $\theta = h(z, v) = z + v$
- We'll have the following figures for $p(v)$, $p(z|\theta)$ and $p(\theta; z|\theta)$:



Example 2

Consider now a localization problem. We wish to know a vehicle pose $X = (x, y)^\top$ thanks to a GPS measurement $z = (x_{gps}, y_{gps})^\top$.

The figure gives $p(z|X)$ and $p(X; z|X)$:



Subsidiary remarks

- since the random value in $p(x;z|x)$ is no longer z but x, then $p(x;z|x)$ is **no longer a *pdf***. Hence :

$$\int p(z|x)dz = 1 \quad \text{but} \quad \int p(x;z|x)dx \neq 1$$

- Actually, $p(z|x)$ is necessary to characterize the sensor having a given value of x but the estimation will require $p(x;z|x)$.

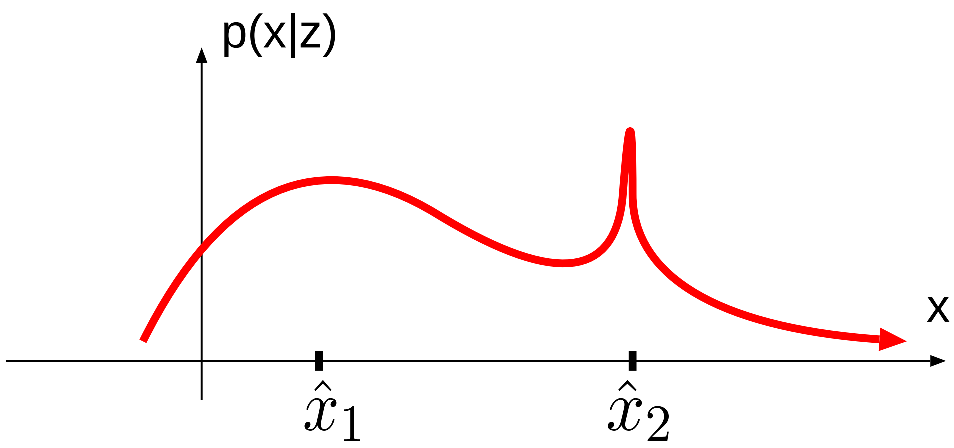
- Now the Bayesian Estimation given in can be rewritten as equation:

$$\boxed{p(x|z) = \frac{p(x;z|x)p(x)}{p(z)}} \tag{3}$$

Estimators

- Even if we know $p(x|z)$ thanks to equation (3), we need more likely a *good* estimation $\hat{x}$ of x.

- But how can we deduce $\hat{x}$ from $p(x|z)$?



Estimator Bias and Variance

Usually we characterize an estimator $\hat{x}$ of x by:

Its bias: $B_{\hat{x}} = \mathbf{E}[\hat{x} - x] = \mathbf{E}[\hat{x}] - x$

- the bias should be null if possible,
- An estimator having a null bias is **unbiased**.

Its variance: $\mathrm{Var}[\hat{x}] = \mathbf{E}\left[(\hat{x} - x)^2\right] = \mathbf{E}\left[\hat{x}^2\right] - \mathbf{E}[x]^2$

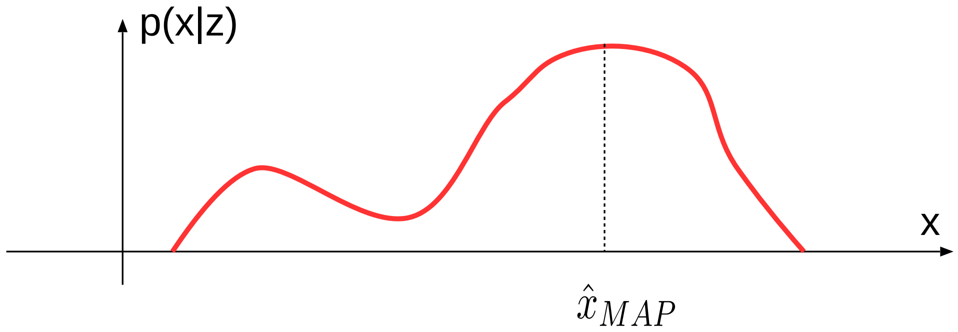
- The variance should be as small as possible,
- The minimum variance of an estimation problem can theoretically be known: it is the **CRLB** (Cramer-Rao Lower Bound) [13],

Usual estimators

MAP estimator

The **MAP** (Maximum A Posteriori) $\hat{x}_{MAP}$ is the value x such as $p(x|z)$ reaches its maximum:

$$\hat{x}_{MAP} = \arg\max_{x} p(x|z) \tag{4}$$

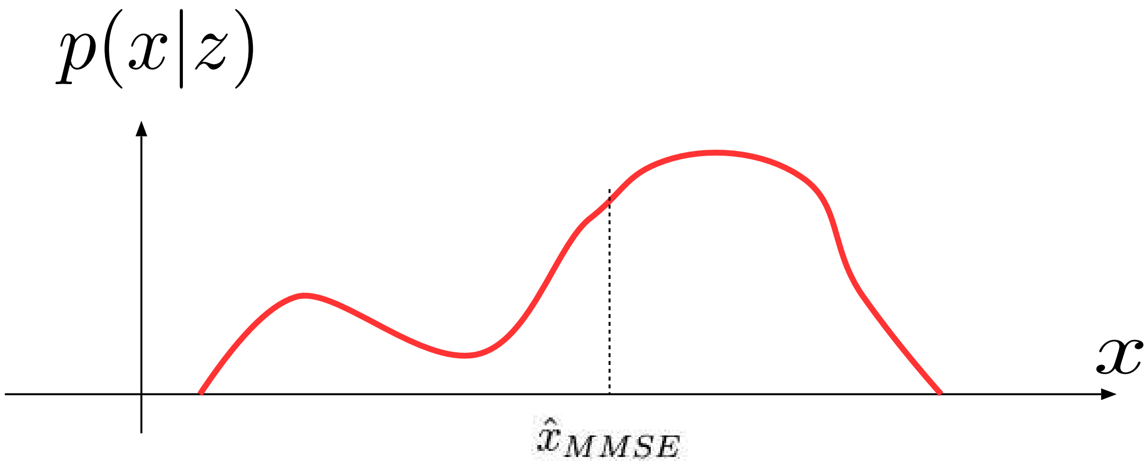


MMSE estimator

The main principle of the **MMSE** (Minimum Mean-Square Error) is to minimize the square errors sum.

$$\hat{x}_{MMSE} = \arg\min_{\hat{x}} \mathbf{E}\left[(\hat{x}-x)^{\top}(\hat{x}-x)|z\right] \tag{5}$$

We can demonstrate that : $\hat{x}_{MMSE} = \mathbf{E}[x|z] = \int xp(x|z)dx$



Bayesian estimators: conclusion

- The behaviour of most of the estimators is approximately the same for symmetrical distributions $p(x|z)$.,
- In the case of multi-modal distributions, the behaviour can lead to *strong errors*.
- Usually the MAP estimator can be easily numerically computed,
- In the case of the MAP and the MMSE, the denominator $p(z)$ of equation (3) does no longer matter since it doesn't depend on x. So for example for the MAP estimator will be:

$$\boxed{\hat{x}_{MAP} = \arg\max_x p(x|z) = \arg\max_x \{p(x; z|x).p(x)\}} \tag{6}$$

Exercise

Consider a scalar measurement z such as $z = \theta + w$ with:

- w: noise such as $w \sim \mathcal{N}(0, \sigma_w^2)$
- the prior knowledge on $\theta \sim \mathcal{N}(\theta_0, \sigma_\theta^2)$.

Determine the Bayesian estimator $\hat{\theta}$ of θ.

Solution

We need first to compute $p(z|\theta)p(\theta)$. Actually, we'll need to find its maximum on θ, this means that we are looking for $p(\theta; z|\theta)p(\theta)$.

Since $z = \theta + w$ and $w \sim \mathcal{N}\left(0, \sigma_w^2\right)$, we therefore have $z \sim \mathcal{N}\left(\theta, \sigma_w^2\right)$:

$$p(z|\theta) = \mathcal{N}\left(\theta, \sigma_w^2\right)$$

and so:

$$p(\theta; z|\theta) = \mathcal{N}\left(\theta, \sigma_w^2\right)$$

We also know:

$$p(\theta) = \mathcal{N}(\theta_0, \sigma_\theta^2)$$

The product of two Gaussian functions $\mathcal{N}(\underline{\mu}_1, \mathbf{C}_1)$ and $\mathcal{N}(\underline{\mu}_2, \mathbf{C}_2)$ will be a Gaussian function given by $\mathcal{N}(\underline{\mu}_1, \mathbf{C}_1) \times \mathcal{N}(\underline{\mu}_2, \mathbf{C}_2) = k\mathcal{N}(\underline{\mu}, \mathbf{C})$

With:

$$\mathbf{C} = \left[\mathbf{C}_1^{-1} + \mathbf{C}_2^{-1}\right]^{-1} \quad \text{and} \quad \mu = \left[\mathbf{C}_1^{-1} + \mathbf{C}_2^{-1}\right]^{-1} \left[\mathbf{C}_1^{-1}\underline{\mu}_1 + \mathbf{C}_2^{-1}\underline{\mu}_2\right]$$

So $p(\theta; z|\theta)p(\theta)$ will be an un-normalized Gaussian function:

$$p(\theta; z|\theta)p(\theta) \propto \mathcal{N}(\mu, \sigma^2) \propto \mathcal{N}(z, \sigma_w^2) \times \mathcal{N}(\theta_0, \sigma_0^2)$$

With:

$$\sigma^2 = \left[\sigma_w^{-2} + \sigma_\theta^{-2}\right]^{-1} = \frac{1}{\sigma_w^{-2} + \sigma_\theta^{-2}} = \frac{\sigma_w^2 . \sigma_\theta^2}{\sigma_w^2 + \sigma_\theta^2}$$

$$\mu = \left[\sigma_w^{-2} + \sigma_\theta^{-2}\right]^{-1} \left[\sigma_w^{-2} z + \sigma_0^{-2}\theta_0\right] = \frac{\sigma_\theta^2 z + \sigma_w^2 \theta_0}{\sigma_w^2 + \sigma_\theta^2}$$

Since the result is a Gaussian function, taking the MAP or the MMSE will lead to the same result: $\hat{\theta} = \mu$

Dynamic Estimation

Introduction

- The goal of the dynamic estimation is to provide an estimation of the **state vector** of a given system.

- The problem here is much more complicated: we have to take into account the evolution on the system.

- We had:

$$p(x|z) = \frac{p(x;z|x)p(x)}{p(z)} \tag{7}$$

- The dynamic estimation will take advantage of that term $p(x)$ that will stem from the previous estimation.

We'll use the following notations:

- x_k: real state vector for time k,
- z_k: measurement achieved for time k
- Z_k: set of measurements achieved until time k.

$$Z_k = \{z_0, \cdots, z_k\} \quad \text{and} \quad Z_{k-1} = \{z_0, \ldots, z_{k-1}\}$$

Actually wish to know x_k taking benefit of all measurements z_k, so:

$$p(x_k | Z_k) = \frac{p(x_k; Z_k | x_k) \times p(x_k)}{p(Z_k)} \tag{8}$$

Stochastic state equations

The State Systems theory provides two equations:

$$\begin{cases} x_k = f(x_{k-1}, u_k, w_k) \\ z_k = h(x_k, v_k) \end{cases} \tag{9}$$

- **The Evolution equation**: defines how the state vector evolves with time and with input u_k and with a **noise evolution** w_k.

  Since x_k is linked to x_{k-1} but not directly to x_{k-2} we use here the **One order Markovian assumption**.

- **The Measurement equation**: is actually the one we saw linking z measurement to x_k state ; v_k is the *observation noise*.

In order to take benefit of the Bayesian formulation, we prefer to use *pdf* representation of both equations in the State System:

$$\begin{cases} x_k = f(x_{k-1}, u_k, w_k) \\ z_k = h(x_k, v_k) \end{cases} \Rightarrow \begin{cases} p(x_k|x_{k-1}) \\ p(x_k; z_k|x_k) \end{cases} \tag{10}$$

Exercise : Consider the linear state model defined by system (11):

$$\begin{cases} x_k = \mathbf{A}x_{k-1} + \mathbf{B}u_k + w_k \\ z_k = \mathbf{C}x_k + v_k \end{cases} \tag{11}$$

- **A**, **B**, **C** are constant matrices and v_k, w_k centred Gaussian noises with covariance matrices: $\mathbf{Cov}[v_k] = \mathbf{R}_k$ and $\mathbf{Cov}[w_k] = \mathbf{Q}_k$.

- Give $p(x_k|x_{k-1})$ and $p(x_k; z_k|x_k)$.

Equations for Dynamic Bayesian Estimation

We had the following equation:

$$p(x_k|Z_k) = \frac{p(x_k; Z_k|x_k) \times p(x_k)}{p(Z_k)} \tag{12}$$

- Our goal is now to provide $p(x_k|Z_k)$ using the previous estimation $p(x_{k-1}|Z_{k-1})$ in order to both use all the measurements Z_k but also to avoid to increase the computational cost after each time k.
- We need therefore to modify eq. (12) in order to take into account not $p(x)$ but this previous estimation $p(x_{k-1}|Z_{k-1})$.
- Actually we'll split the problem in two parts: **Prediction** and **Updating**.

Prediction

Starting from $p(x_{k-1}|Z_{k-1})$ we'll try to obtain $p(x_k|Z_{k-1})$: this is the **prediction** of x_k without the last measurement z_k.

For two random variables x and y we have:

$$p(x) = \int_{-\infty}^{+\infty} p(x,y)dy \quad \text{and} \quad p(x,y) = p(x|y)p(y)$$

So:

$$p(x_k|Z_{k-1}) = \int p(x_k, x_{k-1}|Z_{k-1})dx_{k-1}$$

Hence:

$$p(x_k|Z_{k-1}) = \int p(x_k|x_{k-1}, Z_{k-1})p(x_{k-1}|Z_{k-1})dx_{k-1}$$

Since all the past of x_k is stored in x_{k-1} (*Markovian assumption*), Z_{k-1} doesn't bring any news to x_k, so:

$$p(x_k|Z_{k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|Z_{k-1})dx_{k-1} \tag{13}$$

This the **Chapman-Kolmogorov** relation [19]. We can notice this equation provides the prediction x_k having all the data until $k-1$ and taking into account:

- the *evolution model* $p(x_k|x_{k-1})$ (see eq. (10),
- the **previous estimation** $p(x_{k-1}|Z_{k-1})$ in a **recursive** process as we wanted.

Updating

Having $p(x_k|Z_{k-1})$ we'll try to get $p(x_k|Z_k)$: here we'll take into account the last measurement z_k to provide the wished *pdf* $p(x_k|Z_k)$. Since $Z_k = \{z_0, \cdots, z_k\}$ and $Z_{k-1} = \{z_0, \ldots, z_{k-1}\}$, we'll have:

$$p(x_k|Z_k) = p(x_k|z_0, \ldots, z_k) = \frac{p(z_k|x_k, z_0, \ldots, z_{k-1})p(x_k|z_0, \ldots, z_{k-1})}{p(z_k|z_0, \ldots, z_{k-1})}$$

We need to make here a strong assumption: **all the measurements z_k are statistically independent**. Then:

$$p(z_k|x_k, z_0, \ldots, z_{k-1}) = p(z_k|x_k) \quad \text{and} \quad p(z_k|z_0, \ldots, z_{k-1}) = p(z_k)$$

So:

$$p(x_k|z_0,\dots,z_k) = \frac{p(z_k|x_k)p(x_k|z_0,\dots,z_{k-1})}{p(z_k)} \propto p(z_k|x_k)p(x_k|z_0,\dots,z_{k-1})$$

And so:

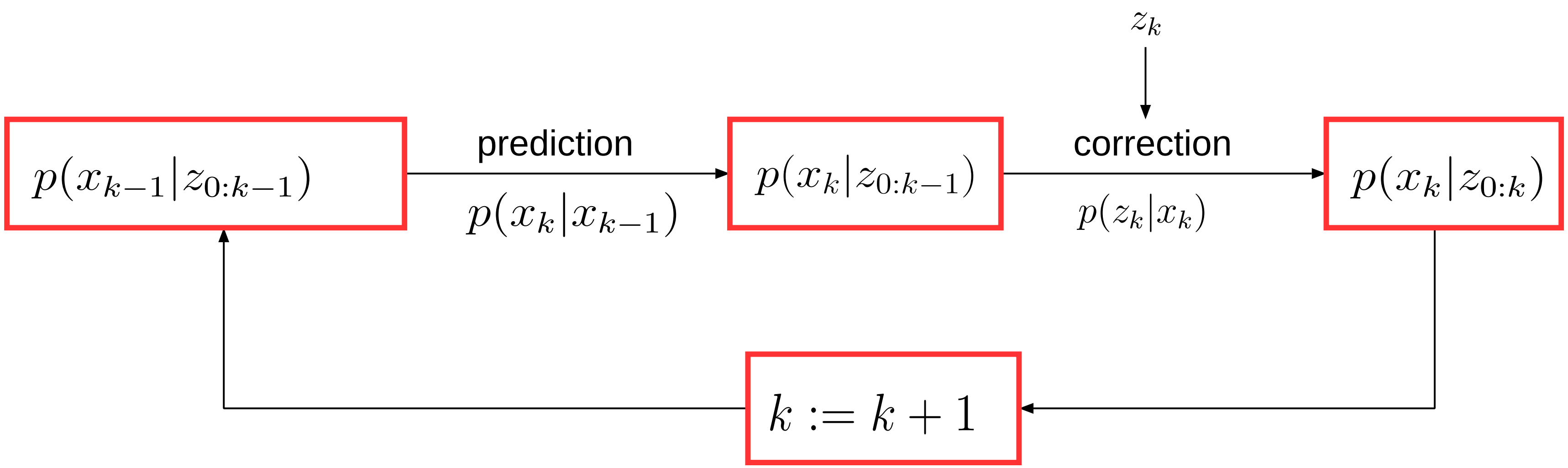
$$p(x_k|Z_k) \propto p(z_k|x_k)p(x_k|Z_{k-1})$$

Since we'll need to optimize this equation regarding x_k we'll need to use the likelihood function $p(x_k;z|x_k)$ and therefore:

$$\boxed{p(x_k|Z_k) \propto p(x_k;z_k|x_k)p(x_k|Z_{k-1})} \tag{14}$$

- (13) makes it possible to update the estimation having the last measurement z_k and using the prediction equation,
- In a same way eq. (14) will be the input pour next time $k+1$ equation (13).

Hence, we obtain a **recursive** algorithm that has a constant computational load:



Bayesian Estimation: subsidiary remarks

- We made two assumptions:
  - the *one order Markov assumption*, that can always be satisfied by choosing a suitable state vector x
  - the independence of the random noises v_k and w_k. This last assumption can be an issue however
- We need to use an estimator to provide the estimation $\hat{x}_k$ of x_k,
- Making a linear assumption for f and h functions in eq (9) and white, independent and Gaussian assumptions for w_k and v_k noises will provide a tractable solution: this is the **Kalman Filter**.

Kalman filtering

- The main goal of the Kalman filter is to provide a solution to dynamic estimation equations (13) and (14) assuming several hypothesis.

- The Kalman filter has been developed by Kalman [11] for discrete time then by Kalman and Bucy [12] for continuous time [8].

The Kalman filter follows the same steps than the generic Bayesian Estimation:

- **Prediction** of the future state value,

- **Estimation** of the current state value.

Linear Stochastic Modelling

The general equations of a stochastic state system are:

$$\begin{cases} x_k = f(x_{k-1}, u_k, w_k) \\ z_k = h(x_k, v_k) \end{cases} \tag{15}$$

We'll assume here the following linear state system:

$$\begin{cases} x_{k+1} = \mathbf{A}x_k + \mathbf{B}u_k + w_k \\ z_k = \mathbf{C}x_k + v_k \end{cases} \tag{16}$$

- x_k, x_{k-1}: State vector for times k and $k-1$,
- u_k: input vectors, z_k: measurement at time k,

We have noises:

- w_k: additive state noise vector,
- v_k: noise measurement vector.

v_k and w_k noises are supposed to be **Gaussian**, **white** and **uncorrelated**:

$$p(w_k) = \mathcal{N}(0, \mathbf{Q}_k)$$
$$p(v_k) = \mathcal{N}(0, \mathbf{R}_k)$$

- $\mathbf{Q}_k$ and $\mathbf{R}_k$ are the covariance matrices associated to these noises.

Notations

We use the following notations:

- x_k: real value of x for time k,
- $x_{k|k}$: estimation of x_k taking into account measurements until k,
- $x_{k|k-1}$: prediction of x_k with measurements until $k-1$,
- $\mathbf{P}_{k|k} = E[(x_{k|k} - x_k)(x_{k|k} - x_k)^\top]$: *a posteriori* covariance matrix on estimation vector $x_{k|k}$,
- $\mathbf{P}_{k|k-1} = E[(x_{k|k-1} - x_k)(x_{k|k-1} - x_k)^\top]$: *a priori* covariance matrix on prediction vector $x_{k|k-1}$.

Linear Kalman Filter algorithm

1. **Initialization**: we assume a Gaussian initial estimation $p(x_0)$

$$p(x_k) = \mathcal{N}(x_{k|k}, \mathbf{P}_{k|k}) \text{ for } k = 0$$

2. $k = k + 1$

3. **Prediction**: of the Gaussian *pdf* for the next time k:

$$p(x_k | y_0, \ldots, y_{k-1}) = \mathcal{N}(x_{k|k-1}, \mathbf{P}_{k|k-1})$$

4. **Updating**: the state estimation having the measurement y_k

$$p(x_k | y_0, \ldots, y_k) = \mathcal{N}(x_{k|k}, \mathbf{P}_{k|k})$$

5. goto 2

Linear Kalman Filter equations

Evolution Equation

- We wish to compute $p(x_k|y_0, \cdots, y_{k-1}) = \mathcal{N}\left(x_{k|k-1}, \mathbf{P}_{k|k-1}\right)$,

- Since we assume a Gaussian law, we only have to compute $x_{k|k-1}$ and $\mathbf{P}_{k|k-1}$,

- The solution is (see [11, 12, 2]):

$$x_{k|k-1} = \mathbf{A} x_{k-1|k-1} + \mathbf{B} u_k \quad ; \quad \mathbf{P}_{k|k-1} = \mathbf{A} \mathbf{P}_{k-1|k-1} \mathbf{A}^\top + \mathbf{Q}_{k-1}$$

Notice these relationships are **recursive**.

Updating Equation

- We wish now to estimate the Gaussian law $p(x_k|y_0,\cdots,y_k)=\mathcal{N}\left(x_{k|k},\mathbf{P}_{k|k}\right)$
- so we only need to compute $x_{k|k}$ and its covariance matrix $\mathbf{P}_{k|k}$.
- The solution is:

$$x_{k|k} = x_{k|k-1} + \mathbf{K}_k(z_k - \mathbf{C}x_{k|k-1})$$

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{C})\,\mathbf{P}_{k|k-1}$$

$\mathbf{K}_k$ is the Kalman gain:

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{C}^\top\left[\mathbf{C}\mathbf{P}_{k|k-1}\mathbf{C}^\top + \mathbf{R}_k\right]^{-1}$$

Linear Kalman Filter Equations

The equations are summarized here:

$$\begin{cases} x_{k|k-1} &= \mathbf{A}x_{k-1|k-1} + \mathbf{B}u_k \\ \mathbf{P}_{k|k-1} &= \mathbf{A}\mathbf{P}_{k-1|k-1}\mathbf{A}^\top + \mathbf{Q}_k \\ x_{k|k} &= x_{k|k-1} + \mathbf{K}_k(z_k - \mathbf{C}x_{k|k-1}) \\ \mathbf{P}_{k|k} &= (\mathbf{I} - \mathbf{K}_k\mathbf{C})\mathbf{P}_{k|k-1} \\ \mathbf{K}_k &= \mathbf{P}_{k|k-1}\mathbf{C}^\top(\mathbf{C}\mathbf{P}_{k|k-1}\mathbf{C}^\top + \mathbf{R}_k)^{-1} \end{cases} \tag{17}$$

Remark: $(z_k - \mathbf{C}x_{k|k-1})$ is called *Innovation*, and this term is weighted by the *Kalman gain* $\mathbf{K}$,

Exercise

We wish to estimate the pose and speed of a vehicle running on a road. This vehicle is seen by a camera embedded in a satellite.

- From the images we can extract the position of the vehicle with a 2 pixels standard deviation Gaussian error.

- The position (u, v) in the image is linked to vehicle position (x, y) by: $u = e_u \frac{x}{h}$ and $v = e_v \frac{y}{h}$, (h: satellite altitude, e_u and e_v: constants).

- We consider a sampling time $T_s = 1s$ and the vehicle speed is approximately constant (with a 1km/h/s standard dev. error).

Determine the Kalman filter parameters allowing to solve this problem.

Evolution

We have : $\underline{X} = (x, \dot{x}, y, \dot{y})^\top$ and a constant speed model, so :

$$\underline{x}_{k+1} = \begin{pmatrix} 1 & T_s & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{pmatrix} + \begin{pmatrix} \underline{w}_{kx} \\ \underline{w}_{ky} \end{pmatrix} \text{ with } \underline{w}_{kx} = \begin{pmatrix} w_{kx} \\ w_{k\dot{x}} \end{pmatrix} \text{ and } \underline{w}_{ky} = \begin{pmatrix} w_{ky} \\ w_{k\dot{y}} \end{pmatrix}$$

Noises on x and y are assumed to be uncorrelated, so we have :

$$\mathbf{Q} = \begin{pmatrix} \mathbf{C}_{kx} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{ky} \end{pmatrix} \quad \text{with} \quad \mathbf{C}_{kx} = \mathbf{C}_{ky} = \sigma_w^2 \begin{pmatrix} \frac{T_s^3}{3} & \frac{T_s^2}{2} \\ \frac{T_s^2}{2} & T_s \end{pmatrix}$$

With $\sigma_w = 1km/h/s = \frac{1000}{3600} = 0.27m/s^2$.

Updating

We know that :

$$\hat{u} = e_u \frac{x}{h} + \epsilon_u \quad \text{and} \quad \hat{v} = e_v \frac{y}{h} + \epsilon_v$$

We will have :

$$z = \begin{pmatrix} \hat{u} \\ \hat{v} \end{pmatrix} = \mathbf{C}\underline{X} + \underline{v}_k \quad \text{with} \quad \underline{v}_k = \begin{pmatrix} \epsilon_u \\ \epsilon_v \end{pmatrix} \quad \text{and} \quad \mathbf{C} = \begin{pmatrix} \frac{e_u}{h} & 0 & 0 & 0 \\ 0 & 0 & \frac{e_v}{h} & 0 \end{pmatrix}$$

Since $\hat{u}$ and $\hat{v}$ are assumed to be uncorrelated and having a 2 pixels noise, we can write :

$$\mathbf{R} = \begin{pmatrix} \sigma_u^2 & 0 \\ 0 & \sigma_v^2 \end{pmatrix} = \begin{pmatrix} 4 & 0 \\ 0 & 4 \end{pmatrix}$$

Linear Kalman filter: remarks

- In the case where the Gaussian hypothesis is not verified, the filter provides a sub-optimal estimation,
- The Kalman Filter can cope with non-stationnary noises,
- However the **white noise constraint** (on both v_k and w_k) can be a problem prone to lead to over-convergences,
- The computational cost of the filter is due mainly to the matrix inversion (size $N \times N$, N is z size),
- The linearity constraint is probably the main issue,
- Several approaches exist: EKF, UKF, Particles Filters, etc.

Extented Kalman Filter

Introduction

- In order to solve the estimation problem even in non-linear, we need to *linearize* the equations.

- The general stochastic state equations are:

$$\begin{cases} x_k = f(x_{k-1}, u_k, w_k) \\ z_k = h(x_k, v_k) \end{cases} \tag{18}$$

- Since h and f are nonlinear functions, matrices **A**, **B** and **C** disappear.

Non linear functions mean and variance

- Consider a random vectorial value $x = (x_1, \cdots, x_N)^\top$ with expected value $\mu_x = \mathbf{E}[x]$ and with covariance $\mathbf{Cov}[x] = \mathbf{C}_x$,

- Consider the following vectorial equation:

$$y = \begin{pmatrix} y_1 \\ \vdots \\ y_M \end{pmatrix} = f(x) = \begin{pmatrix} f_1(x_1, \cdots, x_N) \\ \vdots \\ f_M(x_1, \cdots, x_N) \end{pmatrix}$$

- What are the expected value $\mathbf{E}[y]$ and the covariance $\mathbf{Cov}[y]$?

Expected value of y: We can write $x = \mu_x + \epsilon_x$.

Here, ϵ_x is stochastic part of x such as $\mathbf{E}[\epsilon_x] = 0$ and $\mathbf{Cov}[\epsilon_x] = \mathbf{C}_x$.

$$y = f(\mu_x + \epsilon_x) \approx f(\mu_x) + \left.\frac{\partial f}{\partial x}\right|_{x=\mu_x} \epsilon_x$$

With the **Jacobian Matrix** $\mathbf{J}_{fx}$:

$$\mathbf{J}_{fx} = \left.\frac{\partial f}{\partial x}\right|_{x=\mu_x} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_N} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_M}{\partial x_1} & \cdots & \frac{\partial f_M}{\partial x_N} \end{pmatrix}$$

We have: $y \approx f(\mu_x) + \mathbf{J}_{fx}\epsilon_x$

So:

$$\mathbf{E}[y] \approx \mathbf{E}[f(\mu_x)] + \mathbf{J}_{fx}\mathbf{E}[\epsilon_x] = f(\mu_x)$$

The Covariance $\mathbf{C}_y$ of y will be (since $f(\mu_x)$ is a constant):

$$\mathbf{Cov}[y] \approx \mathbf{Cov}[\mathbf{J}_{fx}\epsilon_x] = \mathbf{J}_{fx}\mathbf{Cov}[\epsilon_x]\,\mathbf{J}_{fx}^\top$$

So finally:

$$\mathbf{Cov}[y] = \mathbf{C}_y \approx \mathbf{J}_{fx}\mathbf{C}_x\mathbf{J}_{fx}^\top$$

If $z = f(x, y)$ with x and y are **independent**, random values:

$$\mathbf{E}[z] \approx \mathbf{f}(\mu_x, \mu_y) \quad \text{and} \quad \mathbf{Cov}[z] \approx \mathbf{J}_{fx}\mathbf{C}_x\mathbf{J}_{fx}^\top + \mathbf{J}_{fy}\mathbf{C}_y\mathbf{J}_{fy}^\top$$

Linearized Kalman Filter Equations

The prediction equation is given by: $x_k = f(x_{k-1}, u_k, w_k)$.

- Suppose input vector u_k is only given by a measurement $\hat{u}_k$ with centered noise with covariance $\mathbf{C}_{uk}$ then $u_k \sim \mathcal{N}(\hat{u}_k, \mathbf{C}_{uk})$

- Suppose w_k and $\hat{u}_k$ are independent.

- the best prediction of x_k will be:

$$x_{k|k-1} = f(x_{k-1|k-1}, \hat{u}_k, 0)$$

- The covariance matrix $\mathbf{P}_{k|k-1}$ of $x_{k|k-1}$ will be:

$$\begin{aligned}\mathbf{P}_{k|k-1} &= \mathbf{J}_{fX} Cov[x_{k-1|k-1}] \mathbf{J}_{fX}^\top + \mathbf{J}_{fu} \mathbf{C}_{uk} \mathbf{J}_{fu}^\top + \mathbf{J}_{fw} Cov[w_k] \mathbf{J}_{fw}^\top \\ &= \mathbf{J}_{fX} \mathbf{P}_{k-1|k-1} \mathbf{J}_{fX}^\top + \mathbf{J}_{fu} \mathbf{C}_{uk} \mathbf{J}_{fu}^\top + \mathbf{J}_{fw} \mathbf{Q}_k \mathbf{J}_{fw}^\top\end{aligned}$$

Updating equations

In the linear case we had:

$$\begin{cases} x_{k|k} & = x_{k|k-1} + \mathbf{K}_k(z_k - \mathbf{C}x_{k|k-1}) \\ \mathbf{P}_{k|k} & = (\mathbf{I} - \mathbf{K}_k\mathbf{C})\mathbf{P}_{k|k-1} \\ \mathbf{K}_k & = \mathbf{P}_{k|k-1}\mathbf{C}^\top(\mathbf{C}\mathbf{P}_{k|k-1}\mathbf{C}^\top + \mathbf{R}_k)^{-1} \end{cases} \quad (19)$$

We know z_k is given by $z_k = h(x_k, v_k)$.

- The estimated value $z_{k|k-1}$ will be approximately given by:

$$z_{k|k-1} = h(x_{k|k-1}, 0)$$

- Using the linear case, $x_{k|k}$ will be:

$$x_{k|k} = x_{k|k-1} + \mathbf{K}_k \left(z_k - h(x_{k|k-1}, 0)\right)$$

- In a similar way, we can linearize observation equation around x_0 and 0 for the centered noise v_k:

$$z_k \approx h(x_0, 0) + \mathbf{J}_{hX}(x_{k-1|k-1} - x_0) + \mathbf{J}_{hv} v_k$$

$\mathbf{J}_{hX}$ and $\mathbf{J}_{hv}$ are the *Jacobian* matrices of fonction h:

$$\mathbf{J}_{hX} = \left.\frac{\partial h}{\partial x}\right|_{x=x_0} \text{ and } \mathbf{J}_{hv} = \left.\frac{\partial h}{\partial v}\right|_{v=0}$$

- By analogy with linear systems, we have:

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{J}_{hX})\mathbf{P}_{k|k-1}$$

and :

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{J}_{hX}^\top(\mathbf{J}_{hX}\mathbf{P}_{k|k-1}\mathbf{J}_{hX}^\top + \mathbf{J}_{hv}\mathbf{R}_k\mathbf{J}_{hv}^\top)^{-1}$$

Linearized Kalman filter : Equations

The solution of the Linearized Kalman filter is:

$$
\begin{cases}
x_{k|k-1} & = f\left(x_{k-1|k-1}, \hat{u}_k, 0\right) \\
\mathbf{P}_{k|k-1} & = \mathbf{J}_{fX}\mathbf{P}_{k-1|k-1}\mathbf{J}_{fX}^\top + \mathbf{J}_{fw}\mathbf{Q}_k\mathbf{J}_{fw}^\top + \mathbf{J}_{fu}\mathbf{C}_{u_k}\mathbf{J}_{fu}^\top \\
x_{k|k} & = x_{k|k-1} + \mathbf{K}_k\left(z_k - h(x_{k|k-1}, 0)\right) \\
\mathbf{P}_{k|k} & = (\mathbf{I} - \mathbf{K}_k\mathbf{J}_{hX})\mathbf{P}_{k|k-1} \\
\mathbf{K}_k & = \mathbf{P}_{k|k-1}\mathbf{J}_{hX}^\top(\mathbf{J}_{hX}\mathbf{P}_{k|k-1}\mathbf{J}_{hX}^\top + \mathbf{J}_{hv}\mathbf{R}_k\mathbf{J}_{hv}^\top)^{-1}
\end{cases}
$$

With:

$$
\mathbf{J}_{fX} = \left.\frac{\partial f}{\partial x}\right|_{x=x_0} \quad , \quad \mathbf{J}_{fw} = \left.\frac{\partial f}{\partial w}\right|_{w=0} , \quad \mathbf{J}_{fu} = \left.\frac{\partial f}{\partial u}\right|_{u=\hat{u}}
$$

$$
\mathbf{J}_{hX} = \left.\frac{\partial h}{\partial x}\right|_{x=x_0} \quad , \quad \mathbf{J}_{hv} = \left.\frac{\partial h}{\partial v}\right|_{v=0}
$$

Extended Kalman Filter

- The main problem of the Linearized Kalman Filter is that the linearization is achieved around a **fixed value** x_0.
- If this value is far from the real one, the linearization leads to errors and even to instabilities of the filter.
- The EKFilter principle is to linearize the state equation (see [16]):
  - around the estimated value $x_{k-1|k-1}$ for prediction step,
  - around the prior estimated value $x_{k|k-1}$ for updating step,
  - around the current measurement $\underline{\hat{u}}_k$

  This involves **the Jacobians computation at each time** k.

EKF Equations

Equations of the Extended Kalman Filter are:

$$\begin{cases} x_{k|k-1} &= f\left(x_{k-1|k-1}, \underline{\hat{u}}_k, 0\right) \\ \mathbf{P}_{k|k-1} &= \mathbf{J}_{fX}\mathbf{P}_{k-1|k-1}\mathbf{J}_{fX}^\top + \mathbf{J}_{fu}\mathbf{C}_{uk}\mathbf{J}_{fu}^\top + \mathbf{J}_{fw}\mathbf{Q}_{wk}\mathbf{J}_{fw}^\top \\ x_{k|k} &= x_{k|k-1} + \mathbf{K}_k\left(z_k - h(x_{k|k-1}, 0)\right) \\ \mathbf{P}_{k|k} &= (\mathbf{I} - \mathbf{K}_k\mathbf{J}_{hX})\mathbf{P}_{k|k-1} \\ \mathbf{K}_k &= \mathbf{P}_{k|k-1}\mathbf{J}_{hX}^\top(\mathbf{J}_{hX}\mathbf{P}_{k|k-1}\mathbf{J}_{hX}^\top + \mathbf{J}_{hv}\mathbf{R}_k\mathbf{J}_{hv}^\top)^{-1} \end{cases} \tag{20}$$

with:

$$\mathbf{J}_{fX} = \left.\frac{\partial f}{\partial x}\right|_{x=x_{k-1|k-1}} \quad , \quad \mathbf{J}_{fu} = \left.\frac{\partial f}{\partial u}\right|_{u=\underline{\hat{u}}_k} \quad , \quad \mathbf{J}_{fw} = \left.\frac{\partial f}{\partial w}\right|_{w=0}$$

$$\mathbf{J}_{hX} = \left.\frac{\partial h}{\partial x}\right|_{x=x_{k|k-1}} \quad , \quad \mathbf{J}_{hv} = \left.\frac{\partial h}{\partial v}\right|_{v=0}$$

Exercice

We want to estimate accurately a vehicle position $X = (x, y, \theta)^\top$ by using a GPS receiver, an odometer and a wheel angle sensor.

- GPS receiver provides the vehicle position estimation $(\hat{X}, \hat{y})$ with an error assumed to be stationary, white, Gaussian and having a σ^2 variance.

- The odometer is assumed to provide an estimation $\hat{ds}$ of the vehicle displacement between k and $k+1$ with a white gaussian error with variance σ_{ds}^2.

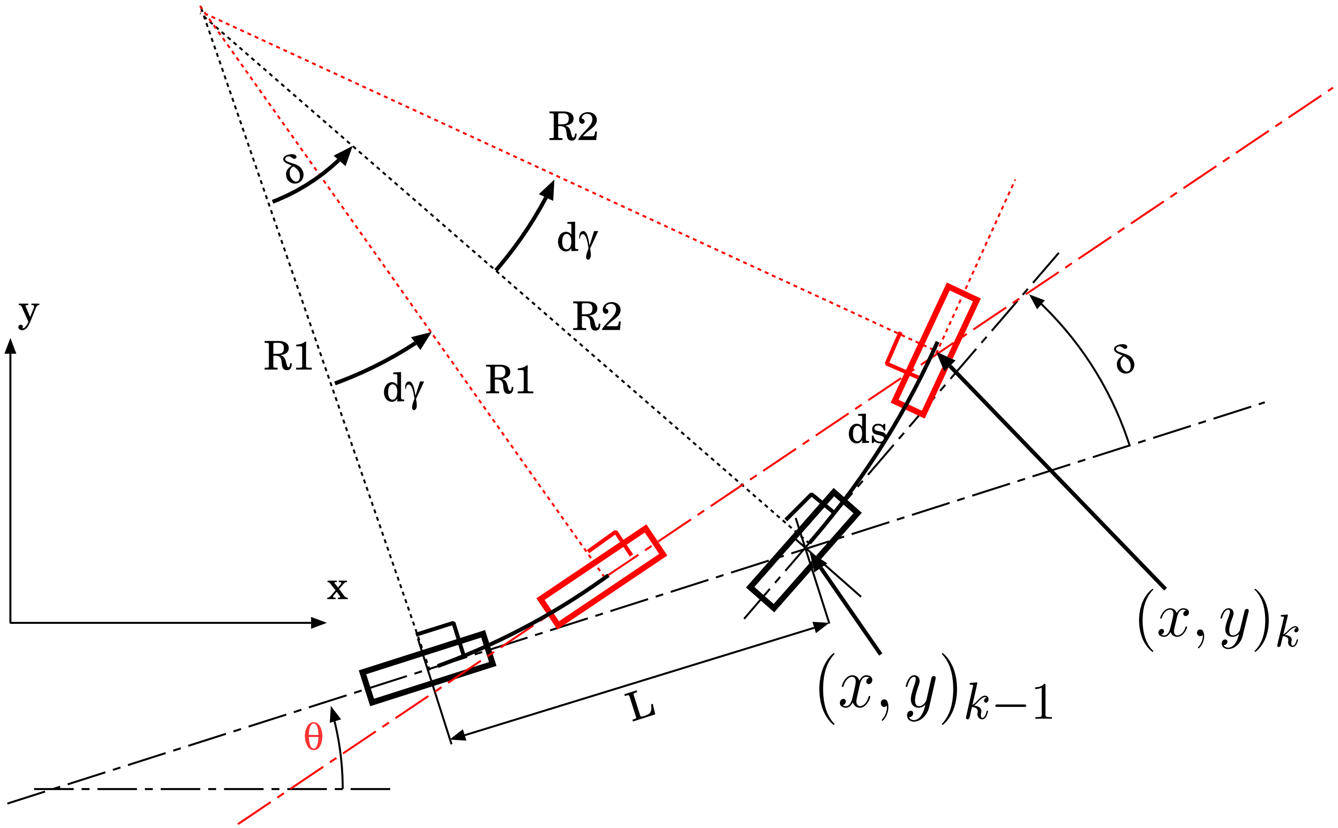
- The wheel angle δ is given by a sensor that gives an estimation $\hat{\delta}$ of δ with a white gaussian error with variance σ_{δ}^2.

- We assume the following Ackermann model:

$$\begin{cases} x_k = x_{k-1} + ds\cos(\theta_{k-1} + d\theta/2) \\ y_k = y_{k-1} + ds\sin(\theta_{k-1} + d\theta/2) \\ \theta_k = \theta_{k-1} + \frac{ds}{L}\sin(\delta) \end{cases} \quad \text{with} \quad d\theta = \frac{ds}{L}\sin\theta$$

L is the distance between vehicle axles.

- Determine the EKF parameters to solve the problem.


R2
δ
dγ
y
R1
dγ
R1
R2
δ
ds
x
$(x, y)_k$
$(x, y)_{k-1}$
L
θ

Solution

Observation equation

- We have $x_{gps} = x + v_{xgps}$ and $y_{gps} = y + v_{ygps}$,

- this is a **linear** equation so:

$$z_k = \begin{pmatrix} x_{gps} \\ y_{gps} \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}}_{\mathbf{C}} \underbrace{\begin{pmatrix} x \\ y \\ \theta \end{pmatrix}}_{\underline{X_k}} + \underbrace{\begin{pmatrix} v_{xgps} \\ v_{ygps} \end{pmatrix}}_{v_k}$$

- We have also:

$$\mathbf{R} = Cov[v_k] = \begin{pmatrix} \sigma^2_{gps} & 0 \\ 0 & \sigma^2_{gps} \end{pmatrix} = \sigma^2_{gps} \mathbf{I}_{2\times 2}$$

Updating equation

- We look for covariance of $x_{k+1|k}$. We'll have:

$$\begin{aligned}\mathbf{P}_{k|k-1} &= Cov[x_{k|k-1}] \\ &= Cov[\mathbf{f}(x_{k-1|k-1}, \hat{ds}, \hat{\delta})] \\ &= \mathbf{J}_{fx}\mathbf{P}_{k-1|k-1}\mathbf{J}_{fx}^{\top} + \mathbf{J}_{ds}\sigma_{ds}^2\mathbf{J}_{ds}^{\top} + \mathbf{J}_{\delta}\sigma_{\delta}^2\mathbf{J}_{\delta}^{\top}\end{aligned}$$

- $\mathbf{J}_{\delta}$ and $\mathbf{J}_{ds}$ are the Jacobian matrices of fonction $\mathbf{f}$ with respect to δ and ds.

- They are:

$$\mathbf{J}_{ds} = \begin{pmatrix} \frac{\partial f_x}{\partial ds} \\ \frac{\partial f_y}{\partial ds} \\ \frac{\partial f_\theta}{\partial ds} \end{pmatrix} = \begin{pmatrix} \cos(\theta + \frac{d\theta}{2}) \\ \sin(\theta + \frac{d\theta}{2}) \\ \frac{\sin\delta}{L} \end{pmatrix} \quad ; \quad \mathbf{J}_{\delta} = \begin{pmatrix} \frac{\partial f_x}{\partial \delta} \\ \frac{\partial f_y}{\partial \delta} \\ \frac{\partial f_\theta}{\partial \delta} \end{pmatrix} = \begin{pmatrix} -\frac{ds^2}{2L}\cos\delta\,\sin\left(\theta + \frac{d\theta}{2}\right) \\ \frac{ds^2}{2L}\cos\delta\,\cos\left(\theta + \frac{d\theta}{2}\right) \\ \frac{ds}{L}\cos(\delta) \end{pmatrix}$$

$$\mathbf{J}_{fx} = \begin{pmatrix} \frac{\partial f_x}{\partial x} & \frac{\partial f_x}{\partial y} & \frac{\partial f_x}{\partial \theta} \\ \frac{\partial f_y}{\partial x} & \frac{\partial f_y}{\partial y} & \frac{\partial f_y}{\partial \theta} \\ \frac{\partial f_\theta}{\partial x} & \frac{\partial f_\theta}{\partial y} & \frac{\partial f_\theta}{\partial \theta} \end{pmatrix} = \begin{pmatrix} 1 & 0 & -ds \sin\left(\theta + \frac{d\theta}{2}\right) \\ 0 & 1 & ds \cos\left(\theta + \frac{d\theta}{2}\right) \\ 0 & 0 & 1 \end{pmatrix}$$

EKF equations

$$\begin{cases} x_{k|k-1} &= f\left(x_{k-1|k-1}, \hat{ds}, \hat{\delta})\right) \\ \mathbf{P}_{k|k-1} &= \mathbf{J}_{fx}\mathbf{P}_{k|k}\mathbf{J}_{fx}^\top + \mathbf{J}_{ds}\sigma_{ds}^2\mathbf{J}_{ds}^\top + \mathbf{J}_\delta\sigma_\delta^2\mathbf{J}_\delta^\top \\ x_{k|k} &= x_{k|k-1} + \mathbf{K}_k\left(z_k - \mathbf{C}x_{k|k-1}\right) \\ \mathbf{P}_{k|k} &= (\mathbf{I} - \mathbf{K}_k\mathbf{C})\mathbf{P}_{k|k-1} \\ \mathbf{K}_k &= \mathbf{P}_{k|k-1}\mathbf{C}^\top(\mathbf{C}\mathbf{P}_{k|k-1}\mathbf{C}^\top + \mathbf{R}_k)^{-1} \end{cases}$$

With:

$$\mathbf{J}_{ds} = \begin{pmatrix} \cos(\theta + \frac{d\theta}{2}) \\ \sin(\theta + \frac{d\theta}{2}) \\ \frac{\sin\delta}{L} \end{pmatrix}, \mathbf{J}_\delta = \begin{pmatrix} -\frac{ds^2}{2L}\cos\delta\sin\left(\theta + \frac{d\theta}{2}\right) \\ \frac{ds^2}{2L}\cos\delta\cos\left(\theta + \frac{d\theta}{2}\right) \\ \frac{ds}{L}\cos(\delta) \end{pmatrix}, \mathbf{C} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \text{ and } d\theta = \frac{ds}{L}\sin\delta$$

Kalman filter conclusions

The EKF is a very powerful tool, nevertheless:

- The filter can diverge due to the linearization errors,
- The *Uncented Kalman Filter* (UKF, see [10]) has been developed to reduce this problem by avoiding high jumps of the linearization point.
- The multi-modalities are not taken into account at all in the EKF.
- Several approaches cope with this problem, mainly: Gaussian Mixtures [21, 1] or Particles filters [3, 6].

Data fusion

Introduction

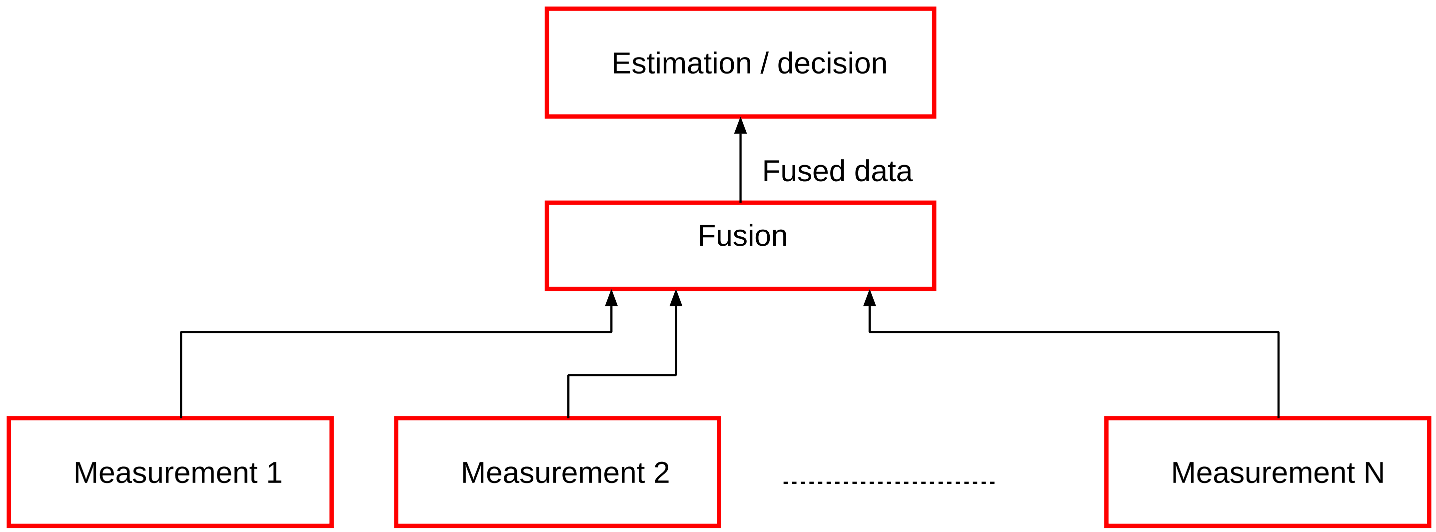
The main goal of data fusion is to combine together several measurements to provide a better result. We expect for instance:

- A better **estimation** (of a robot state for example [9, 22]),
- A **decision** having multiple binary (possibly opposite) measurements [4].
- The target **tracking**. In this case we need to combine both precision and decision having multiple incoming data and multiple state to estimate [4, 15].

We only concentrate **Centralized Architectures** to optimize a state **Estimation** possibly un-synchronized measurements.

Centralized Fusion Architecture

- The centralized fusion aims at combining in the same process the incoming data
- Sometimes Estimation and Fusion are grouped in the same block.
- Sometimes, some data can be cancelled before fusion [5].



Synchronized data

- We suppose here to have N incoming measurements z_{ik} with ($i \in [1, N]$) **at the same time** k.

- Each one of these data z_{ik} is linked to the state vector x_k by the following equation:

$$z_{ik} = h_i(x_k, v_{ik}) \qquad v_{ik} \text{ is the noise on } z_{ik}$$

- Usually the fusion is done with a global Kalman filter. Two approaches can be found:
  - **Global Fusion**
  - **On the Fly** fusion.

Global Fusion

Here, all the z_{ik} measurements in a same vector $z_k = (z_{1k}, z_{2k}, \cdots, z_{Nk})^\top$.

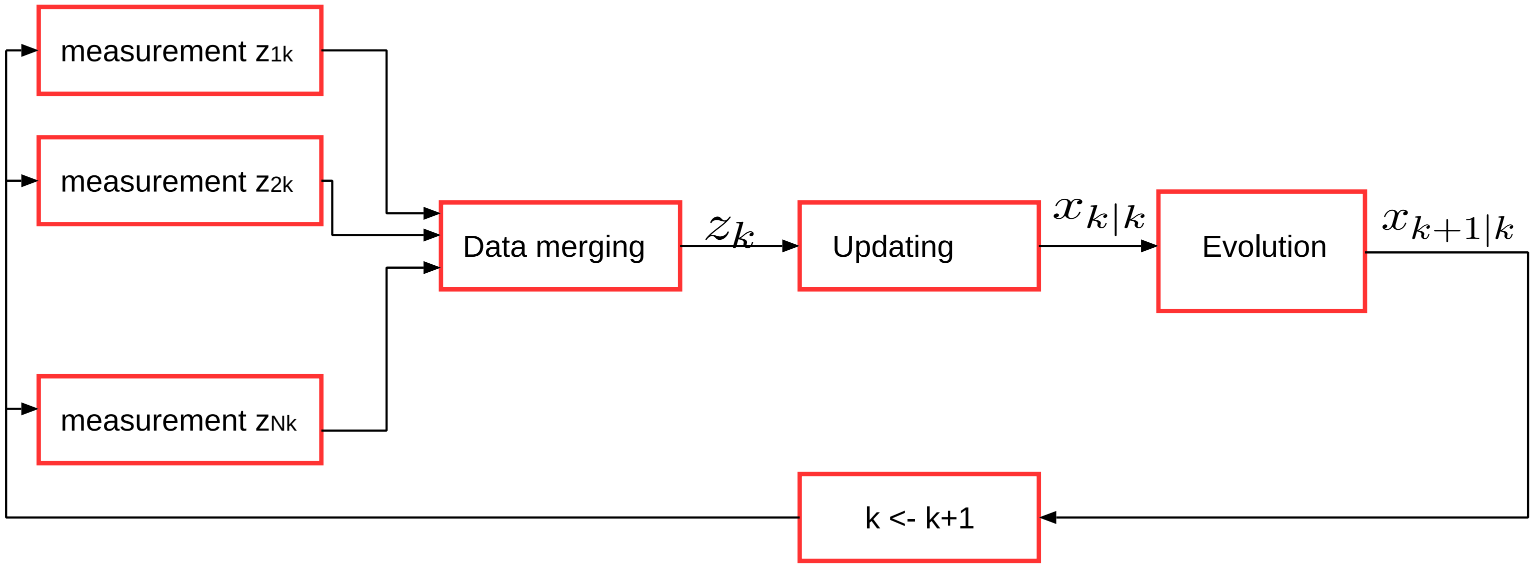
- It will be necessary to define the global covariance matrix $\mathbf{R} = \mathbf{Cov}[z_k]$

$$\mathbf{R}_k = \begin{pmatrix} \mathbf{R}_{11} & \mathbf{R}_{12} & \ldots & \mathbf{R}_{1N} \\ \mathbf{R}_{21} & \mathbf{R}_{22} & \ldots & \mathbf{R}_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{R}_{N1} & \mathbf{R}_{N2} & \ldots & \mathbf{R}_{NN} \end{pmatrix}_k \qquad (21)$$

- The problem becomes a classical estimation problem.

- Complexity is $O(N^3)$,

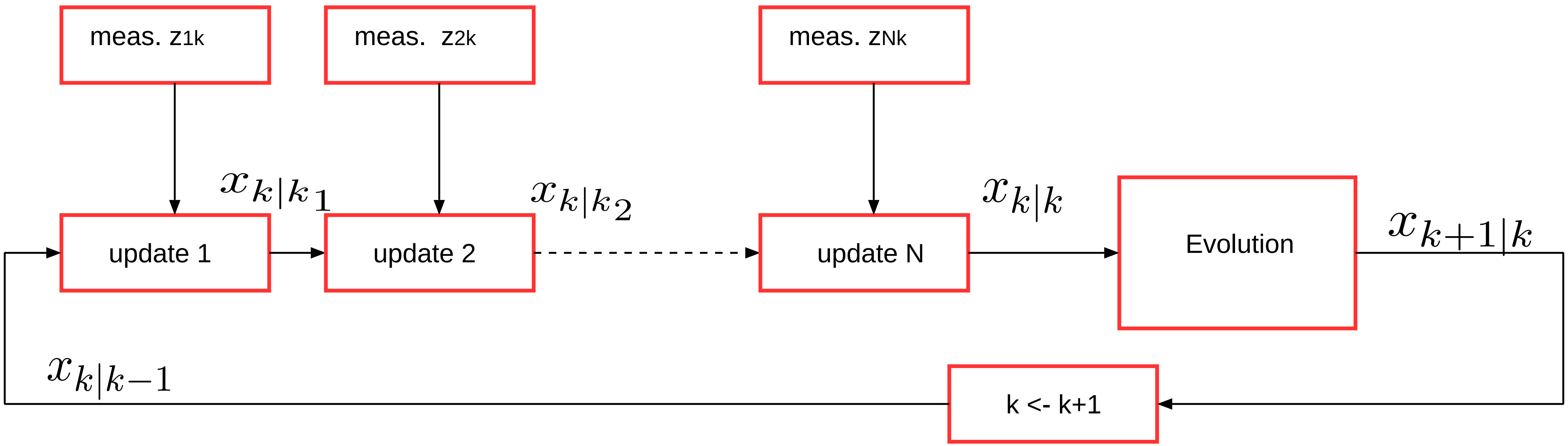
- If measurements are independent, R_k will be diagonal.

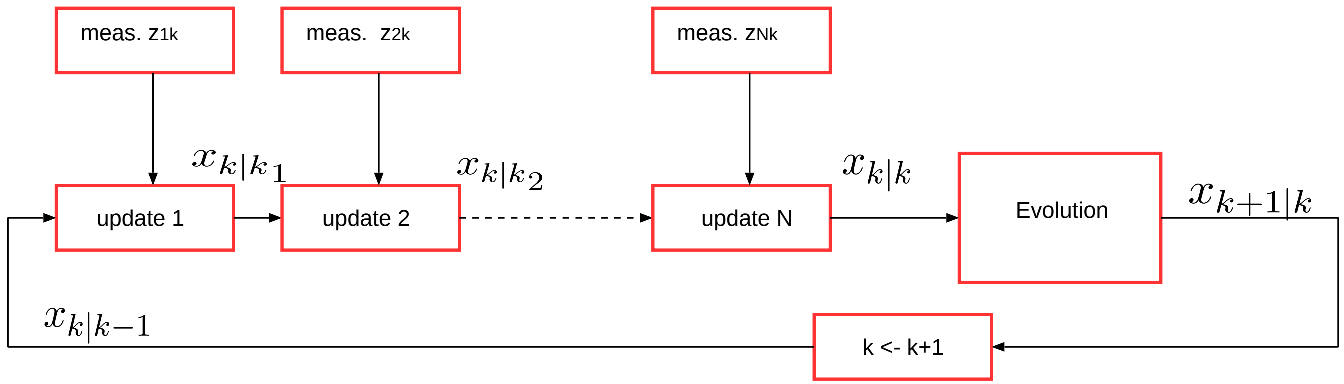


- The main advantage of this solution is that we can take into account all the relationships (correlation) between the data,
- but the computational load is due to the matrix inversion in the Kalman gain.
- Extend vector z_k increases in $O(N^3)$ these times

On the Fly Fusion

We assume z_{ik} data are uncorrelated, and we use the following algorithm:





1. State x_k initialization for time k, (i.e $x_{k|k}$ and $\mathbf{P}_{k|k}$)
2. $k \longleftarrow k+1$
3. Prediction until k (i.e. $x_{k|k-1}$ and $\mathbf{P}_{k|k-1}$ evaluation)
4. For each measurement z_{ik} update state :

   - $x_{k|k_1}$ and $\mathbf{P}_{k|k_1}$ updating with $x_{k-1|k-1}$, $\mathbf{P}_{k-1|k-1}$ and z_{1k},
   - $x_{k|k_2}$ and $\mathbf{P}_{k|k_2}$ updating with $x_{k|k_1}$, $\mathbf{P}_{k|k-1_1}$ and z_{2k},
   - ...
   - $x_{k|k} = x_{k|k_N}$ and $\mathbf{P}_{k|k_N}$ updating with $x_{k|k-1_{N-1}}$, $\mathbf{P}_{k|k-1_{N-1}}$ and z_{Nk},

5. Goto 2

Synchronized fusion: Remarks

- This fusion scheme is **optimal** in the global fusion
- Otherwise, the covariances between z_{ik} measurements are not taken into account: this can lead to sub-optimal estimation or even **over-convergences** is the related noises are correlated.
- In Global Fusion spurious measurements and *non-linearities* are smoothed between the whole data set.
- On the Fly fusion is usually easier to implement, and the computational costs are lower. However no smoothing effect can be done here.
- The main issue to these approaches is that the measurements are required to be **synchronized**. This can be obtained with a prior synchronization step.

Non-synchronized data fusion

- Most of the time, z_{ik} measurements coming from different sensors are **not synchronized**...

- Several solutions exist to solve this issue:

Data synchronization : all the data are interpolated or extrapolated in order to fit the required sampling time.

  - This approach is however sup-optimal most of the cases because of the extra/interpolation errors and the related noise estimation.
  - the latency of the data cannot easily be taken into account.

Delayed processing [22]: data are fused **on the fly**.

Data fusion "on the fly"

This approach aims at solving both the synchronization and the latency problems [22].

- It is worth to distinguish *data* and *measurement*.

  A data d_i is the raw information (an image for instance) taken by the sensor at date t_{di}.

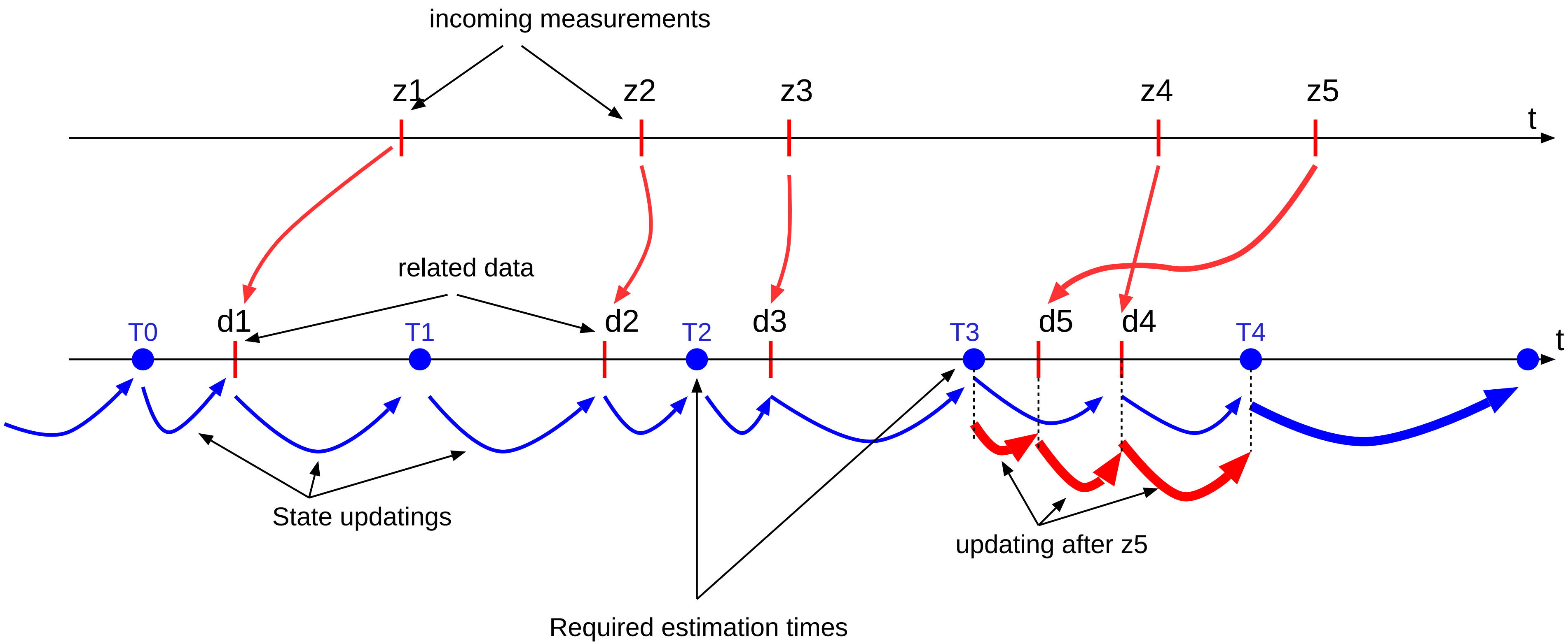
  A measurement z_i is the output of a given processing g_i having data d_i as input. z_i goes in the fusion system at $t_{zi} = t_{di} + \Delta_{ti}$.

- We therefore have: $z_i = g_i(d_i, \Delta_{ti})$

- Δ_{ti} includes the latency of measurement z_i, the processing and the routing times.

- Since sensors have different Δ_{ti} we'll have to face different situations.

- The algorithm presented in [22] is based on an *observation list*.



Suppose we want to estimate periodically the state at each T_i.

- For $t = T_0$ we achieve an evolution step from the last estimation until time T_0 in order to get $\hat{x}_{T0}$.

- At time t_{z1} measurement z_1 gets in the fusion system. Its corresponding data d_1 arrived even before (at time t_{d1}), so z_1 measurement is stored in the observation list with t_{d1} time stamp.

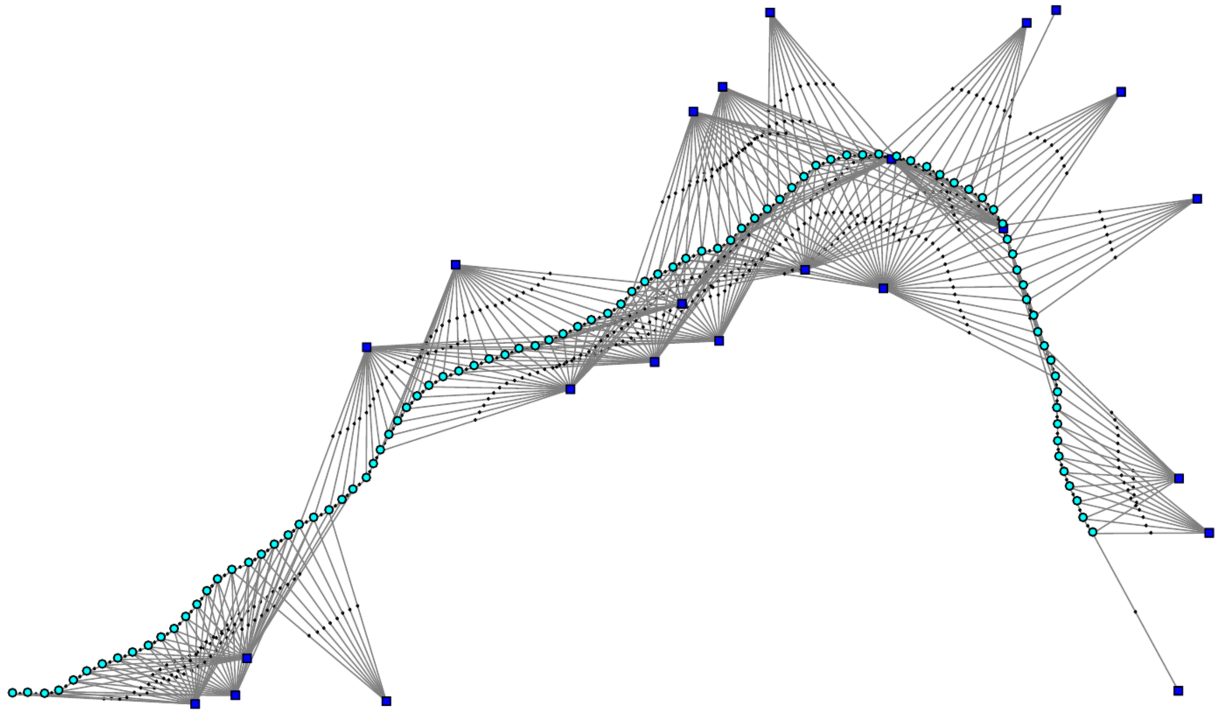
- At $t = T_1$, we need to achieve an evolution step on the estimated state $\hat{x}_{T0}$ until t_{d1}, update this state with measurement z_1, and achieve a new evolution step until T_1 in order to provide $\hat{x}_{T1}$.

The process is iterated an we always update the estimated state starting from **the last state estimation done before the last not yet processed data**.

Data Fusion and Graphical Models

Introduction

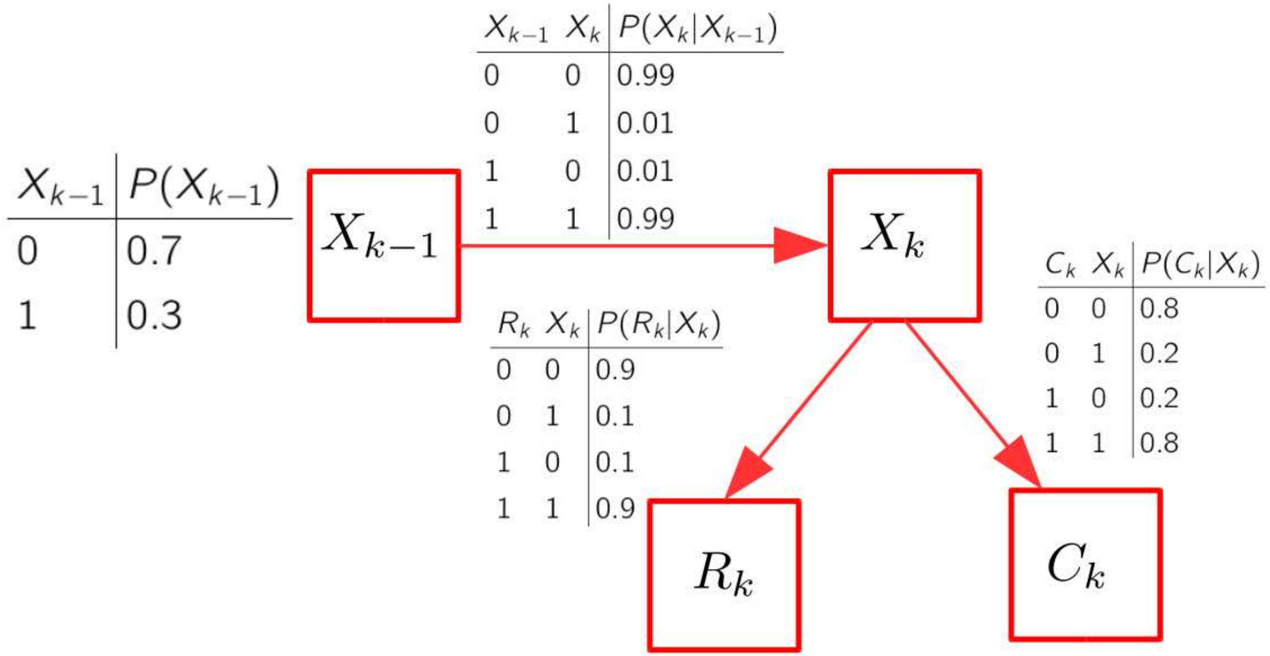
- Sometimes, a graphical representation can be worth to well represent all the actors of the problem and their dependencies.
- It is especially the case for SLAM where the global state can be composed of thousands of poses and landmarks/



Bayes Network

- Bayes Network as been developed by Pearl [17]) and adapted later for dynamic systems [18].
- Suppose, the following *decision* problem:
  - A vehicle embeds a camera and a radar to detect other vehicles ahead.
  - Both camera and radar provides the following binary detection: a vehicle ahead is on our lane on not.
- Let's name $X_k = \{0, 1\}$ the binary event "the vehicle ahead is in our lane" and R_k, C_k the detections of radar and camera.
- The question is: *what is the probability a vehicle ahead is really in our lane ?*

- We can model this problem with a **Bayes Network**: an acyclic graph: nodes represent events and the oriented links represents the joint probability.



X_{k-1}	$P(X_{k-1})$
0	0.7
1	0.3

X_{k-1}	X_k	$P(X_k \mid X_{k-1})$
0	0	0.99
0	1	0.01
1	0	0.01
1	1	0.99

R_k	X_k	$P(R_k \mid X_k)$
0	0	0.9
0	1	0.1
1	0	0.1
1	1	0.9

C_k	X_k	$P(C_k \mid X_k)$
0	0	0.8
0	1	0.2
1	0	0.2
1	1	0.8

- Suppose we got R_k detection but no C_k detection,

- what is $P(X_k | C_k = 0, R_k = 1)$?

- This an *inference* problem: first find the whole *global joint probability* $P(X_{k-1}, X_k, C_k, R_k)$ then we deduce $P(X_k | C_k = 0, R_k = 1)$.

- We can write the **Ancestral rule**: For X_i events ($i \in [1, n]$) we have:

$$\boxed{P(X_1, X_2, \cdots, X_n) = \prod_{i=1}^{n} P(X_i | par(X_i))} \qquad par(X_i)\text{: } X_i \text{ parents} \tag{22}$$

- In our case, we'll get:

$$P(X_{k-1}, X_k, C_k, R_k) = P(X_{k-1}).P(X_k | X_{k-1}).P(C_k | X_k).P(R_k | X_k)$$

- Then, we get the **marginal probability of** $P(X_k, C_k = 0, R_k = 0)$:

$$
\begin{aligned}
P(X_k, C_k = 0, R_k = 0) &= \sum_{X_{k-1}} P(X_{k-1}, X_k, C_k = 0, R_k = 1) \\
&= P(X_{k-1} = 0, X_k, C_k = 0, R_k = 1) \\
&+ P(X_{k-1} = 1, X_k, C_k = 0, R_k = 1)
\end{aligned}
$$

- And finally using to the Bayes Rule yields:

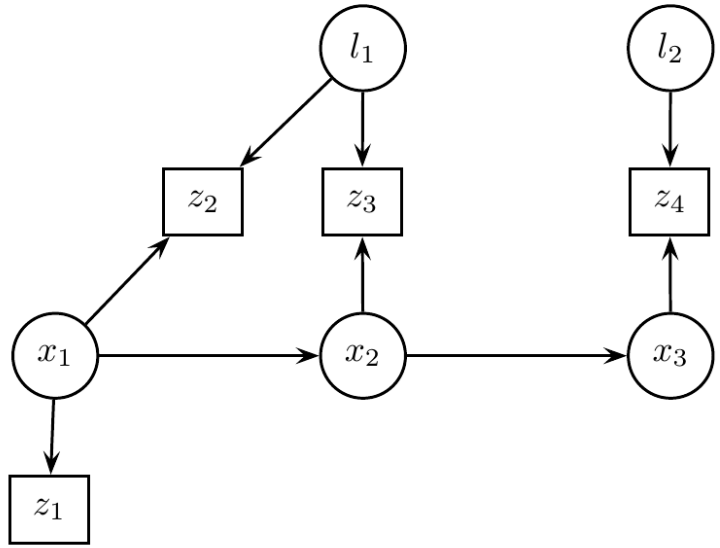
$$P(X_k | C_k, R_k) = \frac{P(X_k, C_k, R_k)}{P(C_k, R_k)}$$

- And so:

$$P(X_k | C_k = 0, R_k = 1) = \frac{P(X_k, C_k = 0, R_k = 1)}{P(C_k = 0, R_k = 1)}$$

Continuous Bayes Network

- RB can manage continuous random variables: we use the *pdf*
- As an example consider the Toy-SLAM [7])



- Here x_k: state value (pose), l_i: landmarks and z_j: measurements
- Let's define $X = (x_1, x_2, x_3, l_1, l_2)$ and $Z = (z_1, z_2, z_3, z_4)$.

Using the *ancestral rule* we can write:

$$
\begin{aligned}
p(X,Z) &= p(x_1).p(x_2|x_1).p(x_3|x_2) && (23)\\
&\times\ p(l_1).p(l_2) && (24)\\
&\times\ p(z_1|x_1) && (25)\\
&\times\ p(z_2|x_1,l_1).p(z_3|x_2,l_1).p(z_4|x_3,l_2) && (26)
\end{aligned}
$$

- eq. (23) is the **Markov chain** linking the pose states x_k,
- eq. (24) is the prior *pdf* on landmarks l_i.
- eq. (25) refers to the link between z_1 and x_1,
- eq (26) represents the relationships between measurements on the landmarks l_i from poses x_k.

- We can write eq (23) to eq. (26) as : $p(X,Z) = p(Z|X).p(X)$ with:

$$\begin{cases} p(X) & = p(x_1).p(x_2|x_1).p(x_3|x_2) \times p(l_1).p(l_2) \\ p(X|Z) & = p(z_1|x_1) \times p(z_2|x_1,l_1).p(z_3|x_2,l_1).p(z_4|x_3,l_2) \end{cases} \tag{27}$$

- Now, we can write the classical Bayes Rule:

$$p(X|Z) = \frac{p(X,Z)}{P(Z)} = \frac{p(Z|X)p(X)}{P(Z)} \tag{28}$$

- It is necessary to consider X as an unknown and Z as known data and so we'll use the **likelihood function** $p(X;Z|X) \stackrel{\Delta}{=} l(X;Z)$:

$$\boxed{p(X|Z) = \frac{p(X;Z|X)p(X)}{P(Z)} \propto l(X;Z)p(X)}$$

MAP estimation

$$p(X|Z) = \frac{p(X;Z|X)p(X)}{P(Z)} \propto l(X;Z)p(X)$$

- This equation provides the *pdf* of both pose states and landmarks.
- Most of the time it is necessary to deduce from $p(X|Z)$ an estimation $\hat{X}$ of X.
- A convenient (and classical) estimator is the **MAP** that can be written here as:

$$\hat{X}_{MAP} = \arg\max_X p(X|Z) = \arg\max_X \{l(X;Z).p(X)\}$$

Factor graphs

From Bayes Networks to Factor Graphs

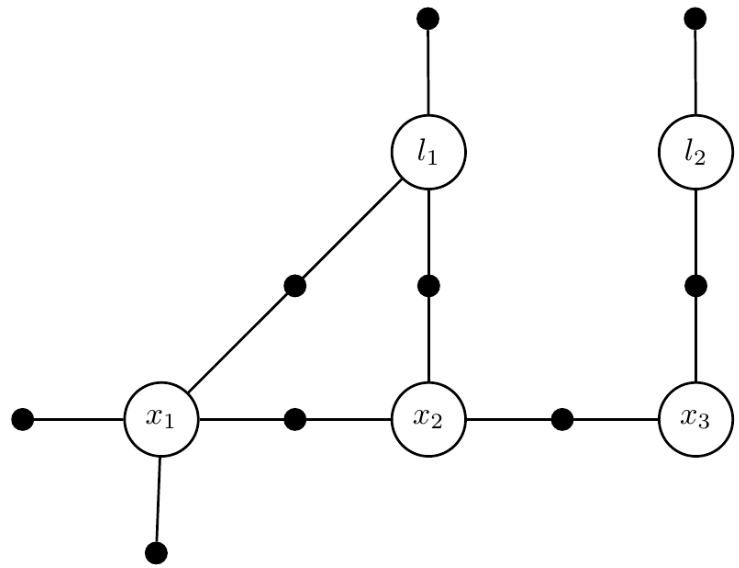
- Since we have $p(X|Z) = \frac{p(X,Z)}{p(Z)}$ we have therefore $p(X|Z) \propto p(X, Z)$.

- We can therefore rewrite equation as:

$$\begin{aligned} p(X|Z) \;\propto\; & p(x_1).p(x_2|x_1).p(x_3|x_2) \\ \times\; & p(l_1).p(l_2) \\ \times\; & l(x_1; z_1) \\ \times\; & l(x_1, l_1; z_2).l(x_2, l_1; z_3).l(x_3, l_2; z_4) \end{aligned} \tag{29}$$

- We have a set of *factors* and in order to make the factorization more clear, we use the **factor graph**.

$$
\begin{aligned}
p(X|Z) \;\propto\;& p(x_1).p(x_2|x_1).p(x_3|x_2) \\
\times\;& p(l_1).p(l_2) \\
\times\;& I(x_1; z_1) \\
\times\;& I(x_1, l_1; z_2).I(x_2, l_1; z_3).I(x_3, l_2; z_4)
\end{aligned}
$$



The 9 big black dots represent the 9 **factors**

- We denote $\phi_k(x_i, x_j)$ the **factor** graph k between nodes x_i and x_j we can define the global factor graph $\phi(X)$ as:

$$\boxed{\phi(X) = \prod_i \phi_i(X_i)}$$

X_i are the set of nodes related to factor ϕ_i.

- Hence we can define the global factor $\phi(l_1, l_2, x_1, x_2, x_3)$ for the toy-SLAM example as:

$$\begin{aligned}\phi(l_1, l_2, x_1, x_2, x_3) &= \phi_1(x_1).\phi_2(x_2, x_1).\phi_3(x_3, x_2) \\ &\times \phi_4(l_1).\phi_5(l_2) \\ &\times \phi_6(x_1) \\ &\times \phi_7(x_1, l_1).\phi_8(x_2, l_1).\phi_9(x_3, l_2)\end{aligned} \tag{30}$$

Inference using Factor graphs

- Having the global factor $\phi(X)$ we look for the estimation $\hat{X}$ of X: Taking the MAP yields:

$$\hat{X}_{MAP} = \arg\max_X \phi(X) = \arg\max_X \prod_i \phi_i(X_i) \tag{31}$$

- Suppose all factors ϕ_i are Gaussian forms:

$$\phi_i(X_i) \propto \exp\left\{-\frac{1}{2}||h(X_i) - z_i||^2_{\mathbf{C}_i}\right\}$$

$$\boxed{\hat{X}_{MAP} = \arg\max_X \phi(X) = \arg\min_X \sum_i \{||h(X_i) - z_i||^2_{\mathbf{C}_i}\}} \tag{32}$$

We therefore solve our global problem by usual numerical minimization.

Remarks

- Factor graphs provide an easy way to solve global fusion problems such as SLAM or others,
- Most of the time factor graphs functions are nonlinear, the minimization requires optimization methods (Gauss-Newton or Levenberg-Marquardt),
- The optimization for visual-SLAM needs to deal with 3D rotations. These specific non linear functions require to use nonlinear manifolds [20, 7]),
- The *sparsity* of the factor graph involves sparse Jacobian matrices in the minimization and leads to very efficient optimizations (see for example the g^2o library [14]).

Bibliography

[1] D.L. Alspach and H.W. Sorenson. Non-linear bayesian estimation using gaussian sum approximation. *IEEE Transaction on Automatica Control*, 17:439–447, 1972.

[2] Brian. D. O. Anderson and John. B. Moore. *Electrical Engineering, Optimal Filtering*. Prentice-Hall Inc., Englewood Cliffs, New Jersey, USA, 1979.

[3] M.S. Arulampalam, S. Maskel, N. Gordon, and T.Clapp. A tutorial on particles filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transaction on Signal Processing*, 50(2):174–188, 2002.

[4] Yaakov Bar-Shalom, Peter K Willett, and Xin Tian. *Tracking and data fusion*. YBS publishing, 2011.

[5] Federico Castanedo. A review of data fusion techniques. *The Scientific World Journal*, 2013, 2013.

[6] D. Crisan and A. Doucet. Survey of convergence results on particle filtering methods for practitioners. *IEEE Transaction on Signal Processing*, 50(3):736–746, 2002.

[7] Frank Dellaert, Michael Kaess, et al. Factor graphs for robot perception. *Foundations and Trends® in Robotics*, 6(1-2):1–139, 2017.

[8] A. Gelb. *Applied Optimal estimation*. MIT press, Cambridge Mass, 1974.

[9] SJ Julier and Jeffrey K Uhlmann. General decentralized data fusion with covariance intersection. *Handbook of multisensor data fusion: theory and practice*, pages 319–344, 2009.

[10] S.J. Julier and J.K. Ulhmann. Unscented filtering and nonlinear estimation. *IEEE Review*, 92(3), Mars 2004.

[11] R. E. Kalman. A new approach to linear filtering and prediction problems. *Trans. ASME, Journal of Basic Engineering*, 82:34–45, 1960.

[12] R. E. Kalman and R. Bucy. A new approach to linear filtering and prediction theory. *Trans. ASME, Journal of Basic Engineering*, 83:95–108, 1961.

[13] S. M. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice Hall, Englewood Cliffs, NJ, 1993.

[14] Rainer Kümmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. g 2 o: A general framework for graph optimization. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3607–3613. IEEE, 2011.

[15] Laetitia Lamard, Roland Chapuis, and Jean-Philippe Boyer. Multi target tracking with cphd filter based on asynchronous sensors. In *International Conference on Information Fusion, Istanbul*, July 2013.

[16] P.S. Maybeck. *Stochastics models, estimation and control*. Academic Press, New York, USA, 1979.

[17] Kevin Murphy. A brief introduction to graphical models and bayesian networks. 1998.

[18] Kevin Patrick Murphy. Dynamic bayesian networks: Representation, inference and learning, 2002.

[19] A. Papoulis. Maximum entropy and spectral estimation: A review. *j-ieee-assp*, 29, 1981.

[20] Geraldo Silveira, Ezio Malis, and Patrick Rives. An efficient direct approach to visual slam. *IEEE transactions on robotics*, 24(5):969–979, 2008.

[21] H.W. Sorenson and D.L. Alspach. Recursive bayesian estimation using gaussian sums. *Automatica*, 7:465–479, 1971.

[22] Cédric Tessier, Christophe Cariou, Christophe Debain, Frédéric Chausse, Roland Chapuis, and Christophe Rousset. A real-time, multi-sensor architecture for fusion of delayed observations: application to vehicle localization. In *Proc. IEEE Intelligent Transportation Systems Conference ITSC '06*, pages 1316–1321, 17–20 Sept. 2006.