Filtering, smoothing and prediction

Sensor fusion & nonlinear filtering

Lars Hammarstrand

WHAT IS FILTERING?

- *Filtering is about recursively estimating parameters of interest based on measurements.*

Notation

- Let $\mathbf{x}_k$ contain parameters of interest and $\mathbf{y}_k$ the measurements at time k. (Time is usually discrete.)

Objective

- Compute $p(\mathbf{x}_k | \mathbf{y}_{1:k})$ where $\mathbf{y}_{1:k} \triangleq \begin{bmatrix} \mathbf{y}_1 & \mathbf{y}_2 & \dots & \mathbf{y}_k \end{bmatrix}$ contains all data up to time k.

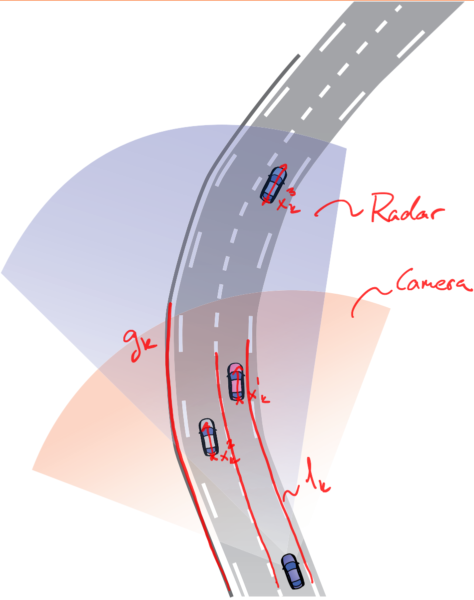
FILTERING IN AUTOMOTIVE APPLICATON

- Vehciles fuses / filters noisy observations from onboard sensor, i.e., radar, lidar and camera, to estimate the current traffic situation:

  $\mathbf{x}_k$: current relative position and velocity of other cars

  $\mathbf{l}_k$: current relative position, headning and shape of the current lane.

  $\mathbf{g}_k$: current relative position, heading and shape of the guard rails.



FILTERING IN OTHER APPLICATIONS

- Historically, positioning of airplanes and ships have been important examples.

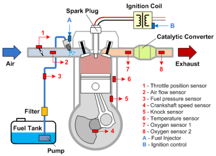
  $\mathbf{x}_k$: positions and velocities of planes

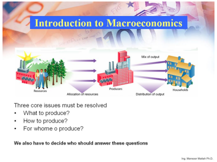
- Control of physical systems often require estimation of the interior state.

  $\mathbf{x}_k$: angle of crankshaft, pressure, etc.

- Often important to assess the states in many other types of systems, e.g., biological or economical.

  $\mathbf{x}_k$: diffusion coefficients, spread of a disease or prices.

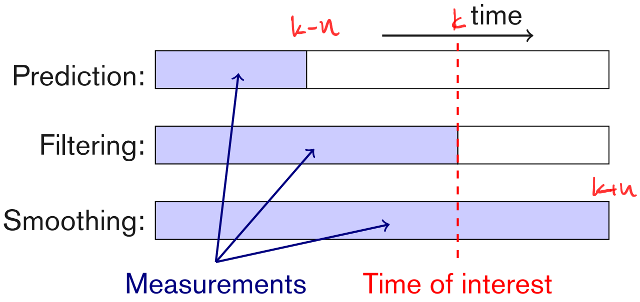




FILTERING, SMOOTHING AND PREDICTION

- Smoothing and prediction are closely related to filtering.

Smoothing ($n > 0$)	**Prediction ($n > 0$)**
• Compute $p(\mathbf{x}_k \mid \mathbf{y}_{1:k+n})$.	• Compute $p(\mathbf{x}_k \mid \mathbf{y}_{1:k-n})$.



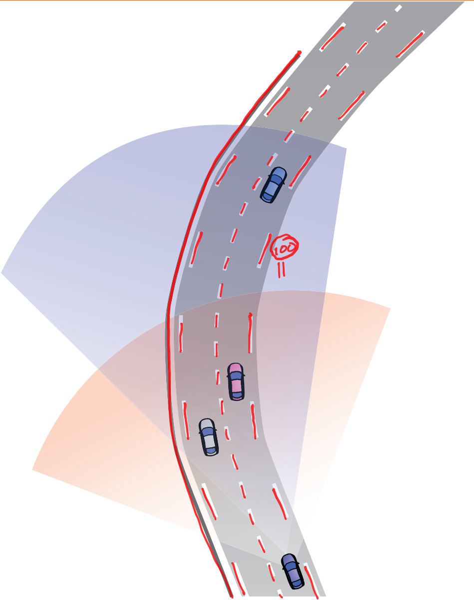
SMOOTHING IN AUTOMOTIVE APPLICATIONS

- Autonmous vehicles use detailed maps to position themselves and to navigate.
- Collect sensor data from many vehicles to jointly estimate their trajectories and the map:

l: global position, headning and shape of the all lanes.

g: global position, heading and shape of the guard rails.

s: global position of signs and its type.

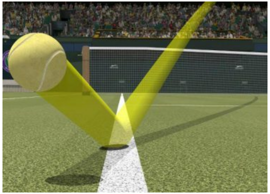


SMOOTHING IN OTHER APPLICATIONS

- Surveillance of, e.g., airports is important for safety reasons.

  $\mathbf{x}_k$: positions of people, bags, etc.

- Other examples:
  - **Communication systems:** having received a complete message you try to decode it.
  - **Sports:** determine where a ball bounced, if someone cheated...
  - **Medicine:** e.g., use sequences of arterial blood pressure to estimate the intracranial pressure.



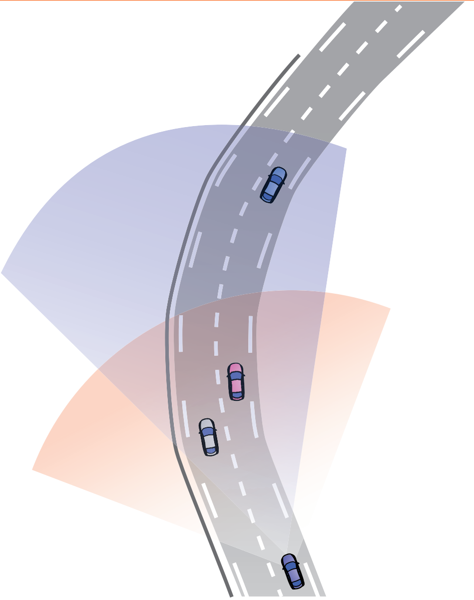
PREDICTIONS IN AUTOMOTIVE APPLICATON

- Vehciles make predicitons of the traffic situation in the near future when, e.g., planning for a safe path or assessing collision risks:

  $\mathbf{x}_{k+n}$: future relative position and velocity of other cars

  $\mathbf{l}_{k+n}$: future relative position, headning and shape of the current lane.

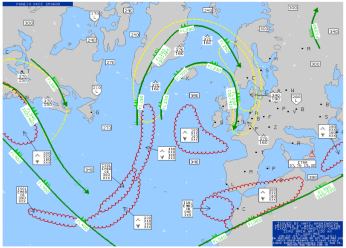
  $\mathbf{g}_{k+n}$: future relative position, heading and shape of the guard rails.



PREDICTION IN OTHER APPLICATIONS

- Weather predictions are important, e.g., to plan routes of airplanes.

  $\mathbf{x}_k$: winds, pressures, temperatures, etc.



- Other examples:
  - **Economy:** the management of companies relies on forecasts of, e.g., demand.
  - **Politics:** many decisions are based on predictions regarding population growth, the financial market, etc.

SELF-ASSESSMENT

Check all that apply.

- The prediction problem is about predicting future measurements given the current state vector.
- In smoothing we conditione on data observed after time k when we compute the distribution of $\mathbf{x}_k$.
- In filtering, smoothing and prediction, both the measurements and the state variables may vary with time.

State space models

Sensor fusion & nonlinear filtering

Lars Hammarstrand

DISCRETE-TIME STATE SPACE MODELS

Discrete-time state space models

For a state vector, $\mathbf{x}_k$, and a measurement vector, $\mathbf{y}_k$, where k denotes a discrete time index, we have the following models,

Motion model:
$$\mathbf{x}_k = f_{k-1}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1}) \tag{1}$$

Measurement model:
$$\mathbf{y}_k = h_k(\mathbf{x}_k, \mathbf{r}_k) \tag{2}$$

where $\mathbf{x}_0 \sim p(\mathbf{x}_0)$.

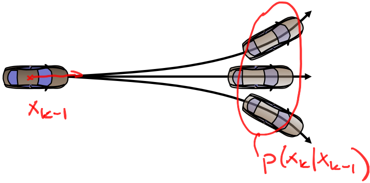
- We also assume that both the motion noise, $\mathbf{q}_{k-1}$, and the measurement noise, $\mathbf{r}_k$, are independent of all other noise vectors.
  - $\rightsquigarrow$ this ensures the Markov property (see the next video).

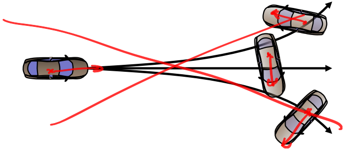
Motion / process model

- The system dynamics are described by (1),

$$\mathbf{x}_k = f_{k-1}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1}), \Leftrightarrow p(x_k | x_{k-1})$$

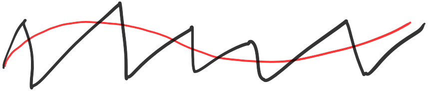
which we refer to as the motion / process model.





Note:

- It describes the state evolution, $p(\mathbf{x}_k | \mathbf{x}_{k-1})$, i.e., the distribution of $\mathbf{x}_k$ given $\mathbf{x}_{k-1}$.
- The motion model thus connects state over time and helps us to rule out unreasonable trajectories.



THE MEASUREMENT MODEL

Measurement model

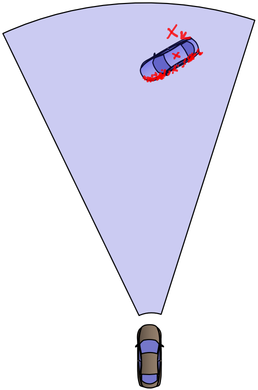
- How the measurements relate to the state vector is described by (2),

$$\mathbf{y}_k = h_k(\mathbf{x}_k, \mathbf{r}_k) \iff p(y_k | x_k)$$

and is called the measurement model or the sensor model.

Note:

- It describes the distribution of $\mathbf{y}_k$ given $\mathbf{x}_k$, $p(\mathbf{y}_k | \mathbf{x}_k)$, i.e., it defines the likelihood function.
- The measurement model relates data to the state vector and helps us to use data to learn about the states.



MODELS WITH INPUT VARIABLES

Known input signal

- The system may also have a known input signal, $\mathbf{u}_k$,

$$\begin{cases} \mathbf{x}_k &= f_{k-1}(\mathbf{x}_{k-1}, \mathbf{u}_k, \mathbf{q}_{k-1}) \\ \mathbf{y}_k &= h_k(\mathbf{x}_k, \mathbf{u}_k, \mathbf{r}_k). \end{cases} \iff \begin{cases} p(\mathbf{x}_k \mid \mathbf{x}_{k-1}; \mathbf{u}_k), \\ p(\mathbf{y}_k \mid \mathbf{x}_k; \mathbf{u}_k), \end{cases}$$

  The time index for $\mathbf{u}$ in the motion model can also be $k-1$.

- The input signal is often a control signal but it may also be an accurate measurement.

SELF-ASSESSMENT

An important benefit with having both a measurement and a motion model is that past data can provide information about the current state.

- True.
- False.

Conditional independencies in state space models

Sensor fusion & nonlinear filtering

Lars Hammarstrand

STATE SPACE MODELS AND CONDITIONAL INDENDENCIES

- We represent state space models in one of two forms:

$$\begin{cases} \mathbf{x}_k &= f_{k-1}(\mathbf{x}_{k-1}, \mathbf{q}_{k-1}) \\ \mathbf{y}_k &= h_k(\mathbf{x}_k, \mathbf{r}_k) \end{cases} \iff \begin{cases} p(\mathbf{x}_k | \mathbf{x}_{k-1}) \\ p(\mathbf{y}_k | \mathbf{x}_k). \end{cases}$$

- For the form on the left hand side we assume:

  Both the motion noise, $\mathbf{q}_{k-1}$, and the measurement noise, $\mathbf{r}_k$, are independent of all other noise vectors.

- The corresponding assumptions for the density representation:

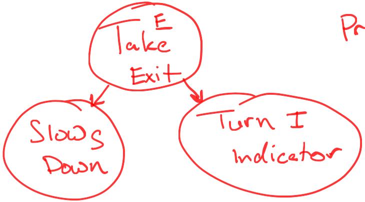
$$p(\mathbf{x}_k | \mathbf{x}_{0:k-1}, \mathbf{y}_{1:k-1}) = p(\mathbf{x}_k | \mathbf{x}_{k-1}) \tag{1}$$

$$p(\mathbf{y}_k | \mathbf{x}_{0:k}, \mathbf{y}_{1:k-1}) = p(\mathbf{y}_k | \mathbf{x}_k) \tag{2}$$

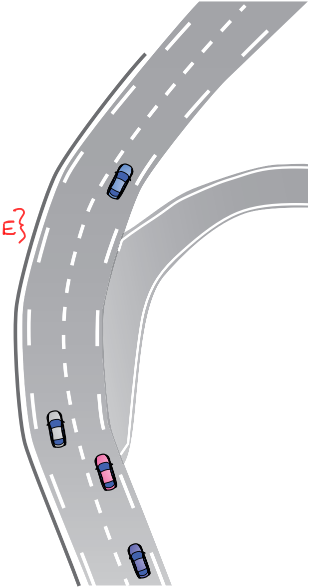
- Note: Both $\mathbf{x}_k$ and $\mathbf{y}_k$ are stochastic processes and the assumption in (1) implies that $\mathbf{x}_k$ is a Markov process.

BAYESIAN NETWORKS AND CONDITIONAL INDEPENDENCIES

- A Bayesian network (also known as belief networks or Bayes net) is a probabilistic graphical model.
- Bayesian networks are directed acyclic graphs that describe how a joint density can be factorized. It also illustrates conditional dependencies.

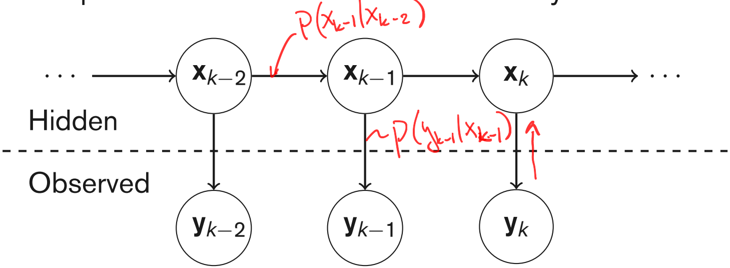


$$\Pr\{S,I,E\} = \Pr\{S|\cancel{I},E\}\Pr\{I|E\}\Pr\{E\}$$
$$= \Pr\{S|E\}\Pr\{I|E\}\Pr\{E\}$$



STATE SPACE MODELS AND BAYESIAN NETWORKS

- A state space model can be described as a Bayesian network:



- The graph illustrates that:

$$
\begin{aligned}
p(\mathbf{x}_{0:k},\mathbf{y}_{1:k}) &= p(\mathbf{y}_{1:k}|\mathbf{x}_{0:k})p(\mathbf{x}_{0:k}) \\
&= p(\mathbf{y}_1|\mathbf{x}_{0:k})p(\mathbf{y}_2|\mathbf{y}_1,\mathbf{x}_{0:k})\dots p(\mathbf{y}_k|\mathbf{y}_{1:k-1},\mathbf{x}_{0:k}) \\
&\quad\; p(\mathbf{x}_0)p(\mathbf{x}_1|\mathbf{x}_0)p(\mathbf{x}_2|\mathbf{x}_{0:1})\dots p(\mathbf{x}_k|\mathbf{x}_{0:k-1})
\end{aligned}
$$

$$= p(y_1|x_1)p(y_2|x_2)\dots p(y_k|x_k)p(x_0)p(x_1|x_0)p(x_2|x_1)\dots p(x_k|x_{k-1})$$

SELF-ASSESSMENT

Suppose the Bayesian network

describes the joint distribution over variables x_{k-1}, x_k and y_k.
Check all that apply:

- $p(x_k, x_{k-1}, y_k) = p(x_k | x_{k-1}) p(x_{k-1}) p(y_k | x_k)$
- $p(x_k, y_k) = p(y_k | x_k) p(x_k)$
- $p(y_k | x_k, x_{k-1}) = p(y_k | x_k)$

Optimal filtering

Sensor fusion & nonlinear filtering

Lars Hammarstrand

FILTERING: PROBLEM FORMULATION

- Consider a time-discrete state space model:

$$p(\mathbf{x}_k \mid \mathbf{x}_{k-1}) \qquad \text{motion model}$$
$$p(\mathbf{y}_k \mid \mathbf{x}_k) \qquad \text{measurement model,}$$

and suppose that $\mathbf{x}_0 \sim p(\mathbf{x}_0)$ and

$$p(\mathbf{x}_k \mid \mathbf{x}_{0:k-1}, \mathbf{y}_{1:k-1}) = p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$$
$$p(\mathbf{y}_k \mid \mathbf{x}_{0:k}, \mathbf{y}_{1:k-1}) = p(\mathbf{y}_k \mid \mathbf{x}_k).$$

Objective in filtering

- We seek to compute $p(\mathbf{x}_k \mid \mathbf{y}_{1:k})$ for $k = 1, 2, 3, \ldots$

A NON-RECURSIVE SOLUTION

- We know Bayesian statistics $\Rightarrow$ we can find $p(\mathbf{x}_k | \mathbf{y}_{1:k})$!

Step 1: use Bayes' rule to find

$$p(\mathbf{x}_{0:k} | \mathbf{y}_{1:k}) = \frac{p(\mathbf{y}_{1:k} | \mathbf{x}_{0:k}) p(\mathbf{x}_{0:k})}{p(\mathbf{y}_{1:k})} \propto p(\mathbf{x}_0) \prod_{i=1}^{k} p(\mathbf{y}_i | \mathbf{x}_i) p(\mathbf{x}_i | \mathbf{x}_{i-1})$$

Step 2: marginalize with respect to $\mathbf{x}_{0:k-1}$

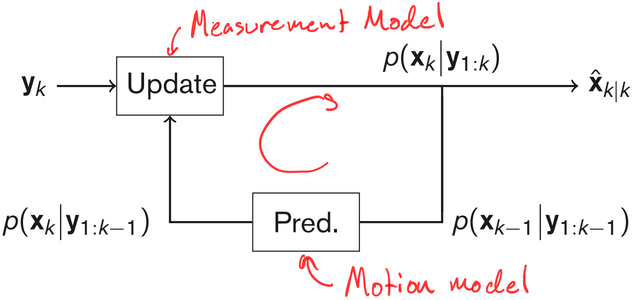
$$p(\mathbf{x}_k | \mathbf{y}_{1:k}) = \int p(\mathbf{x}_{0:k} | \mathbf{y}_{1:k}) \, d\mathbf{x}_{0:k-1}$$

- Weakness: complexity grows with k.

A RECURSIVE FILTERING SOLUTION

Methodology

- Recursively compute $p(\mathbf{x}_k | \mathbf{y}_{1:k})$ from $p(\mathbf{x}_{k-1} | \mathbf{y}_{1:k-1})$.



- A block diagram illustrating the prediction and update steps that we perform recursively.

THE PREDICTION STEP

Prediction

- Compute $p(\mathbf{x}_k | \mathbf{y}_{1:k-1})$ from $p(\mathbf{x}_{k-1} | \mathbf{y}_{1:k-1})$.

- In this step we use our knowledge regarding $\mathbf{x}_{k-1}$, obtained from $\mathbf{y}_{1:k-1}$, to predict $\mathbf{x}_k$.

$$p(\mathbf{x}_k | \mathbf{y}_{1:k-1}) = \int p(x_k, x_{k-1} | y_{1:k-1})\, dx_{k-1} = \int p(x_k | x_{k-1}, \cancel{y_{1:k-1}})\, p(x_{k-1} | y_{1:k-1})\, dx_{k-1}$$

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

This is the *Chapman-Kolmogorov* equation.

SELF-ASSESSMENT ON THE PREDICTION STEP

Suppose $x_k = x_{k-1} + q_k$ where $q_k \sim \mathcal{N}(0, 1)$. The uncertainties in $p(x_k \middle| y_{1:k-1})$ are then normally [select a suitable word below] than the uncertainties in $p(x_{k-1} \middle| y_{1:k-1})$.

- smaller
- larger
- neither larger nor smaller

Only one answer applies.

THE MEASUREMENT UPDATE STEP

Measurement update

- Compute $p(\mathbf{x}_k | \mathbf{y}_{1:k})$ from $p(\mathbf{x}_k | \mathbf{y}_{1:k-1})$.

- In this step, we update our knowledge about $\mathbf{x}_k$ using the new measurement $\mathbf{y}_k$.

$$p(\mathbf{x}_k | \mathbf{y}_{1:k}) = p(x_k | y_k, y_{1:k-1}) = \frac{p(y_k | x_k, \cancel{y_{1:k-1}})\, p(x_k | y_{1:k-1})}{p(y_k | y_{1:k-1})}$$

$$\propto p(y_k | x_k)\, p(x_k | y_{1:k-1})$$

- Note: the prediction and update equations are general.
  They provide a recursive solution to any filtering problem!

SELF-ASSESSMENT ON THE UPDATE STEP

Suppose $y_k = x_k + r_k$ where $r_k \sim \mathcal{N}(0, 1)$. The uncertainties in $p(x_k|y_{1:k})$ are then normally [select a suitable word below] than the uncertainties in $p(x_k|y_{1:k-1})$.

- smaller
- larger
- neither larger nor smaller

Only one answer applies.

OPTIMAL FILTER EXAMPLE

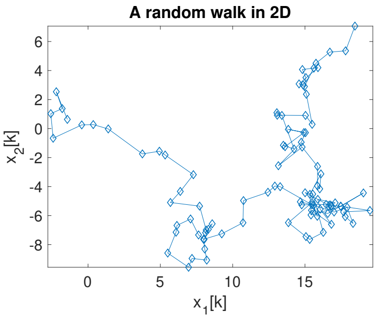
2D random walk with position observations

- Let us consider a 2D state vector, $\mathbf{x}_k = [x_1,\ x_2]^T$, with the following system model

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \mathbf{q}_{k-1} \qquad \mathbf{q}_{k-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$$

$$\mathbf{y}_k = \mathbf{x}_k + \mathbf{r}_k \qquad \mathbf{r}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$$

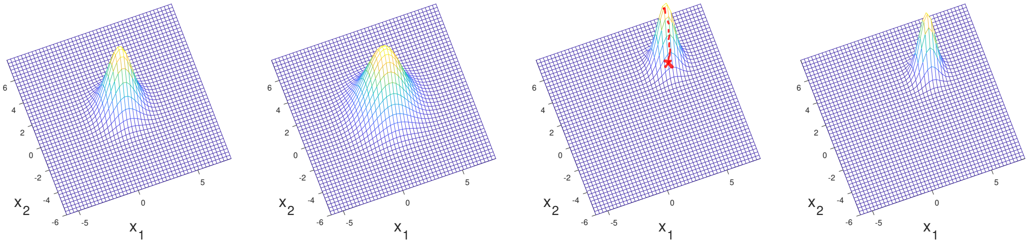
and $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{P}_0)$



PREDICTION AND UPDATE ILLUSTRATIONS

- Optimal filter recursion:

Prior | Predicted density | Likelihood | Updated density



$p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})$ $\quad$ $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$ $\times$ $p(\mathbf{y}_k|\mathbf{x}_k)$ $\propto$ $p(\mathbf{x}_k|\mathbf{y}_{1:k})$

- Note 1: uncertainties increase during prediction step.
- Note 2: posterior $\propto$ prior (predicted density) x likelihood.