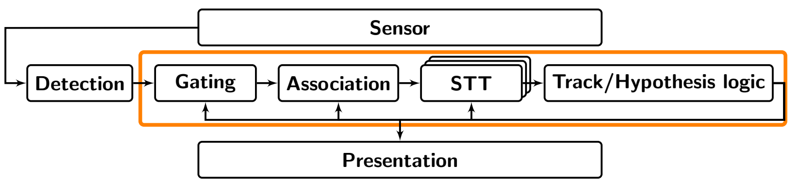
Target Tracking
Le 2: Models in Target Tracking

Gustaf Hendeby and Rickard Karlsson

Div. Automatic Control
Dept. Electrical Engineering
`gustaf.hendeby@liu.se`,
`rickard.g.karlsson@liu.se`

Summary: lecture 1



- Multi-target tracking is the problem of decide how many targets are present and how they move, given measurements with imperfections.
- Classic MTT can be divided in several stages: gating, association, single target tracking, track/hypothesis logic, and presentation.
- Single target tracking: Kalman type filters, particle filters

References on Tracking Models and IMM

- R. X. Li and V. P. Jilkov. Survey of maneuvering target tracking. Part I: Dynamic models.
  IEEE Transactions on Aerospace and Electronic Systems, 39(4):1333–1364, Oct. 2003.
- X. R. Li and V. P. Jilkov. A survey of maneuvering target tracking—Part III: Measurement models.
  In O. E. Drummond, editor, *Proceedings of SPIE — The International Society for Optical Engineering*, volume 4473, Orlando, FL, USA, July 2001.
- X. R. Li and V. P. Jilkov. A survey of maneuvering target tracking—Part V: Multiple-model methods.
  IEEE Transactions on Aerospace and Electronic Systems, 41(4):1255–1321, Oct. 2005.
- H. A. P. Blom and Y. Bar-Shalom. The interacting multiple model algorithm for systems with Markovian switching coefficients.
  IEEE Transactions on Automatic Control, 33(8):780–783, Aug. 1988.
- T. Kirubarajan and Y. Bar-Shalom. Kalman filter versus IMM estimator: When do we need the latter?
  IEEE Transactions on Aerospace and Electronic Systems, 39(4):1452–1457, Oct. 2003.

Overview: models in target tracking

Models

- Consider a model of the target state x_t with (target) input u_t.

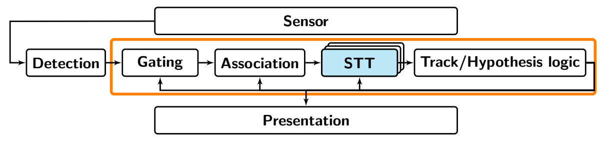
$$x_{t+1} = f(x_t, u_t)$$
$$y_t = h(x_t) + e_t$$

- The input signal, u_t, is unknown (pilot maneuver, external influences, etc)
- We need to replace it with a random noise
- All models are approximations, that might be of high or low fidelity

Hence, one way to model this is to introduce process noise w_t. The measurement noise, e_t, is basically given by the sensor!

Today: common models and maneuvering filters.

Measurements and Measurement Models



Measurements

Measurement Sources

- Previously observed targets
- New targets
- Clutter (false alarms/detections/observations)

Kinematic measurements

- Position (pixel indices)
- Range
- Range rate (radar Doppler shift)
- Bearing

Attribute measurements

- Signal strength
- Intensity
- Aspect ratio
- Target type

We will only talk about kinematic measurements!

Measurement Scan

In many applications data is received during some time period, a scan. For example a scanning radar (*e.g.*, $f = 1$ Hz) receives all measurements for one revolution once the full revolution is finished.

Typically, if the targets do not move too fast, tracking can be performed assuming all the measurements in one scan are obtained at the same time.

False Measurements: clutter

- A false measurement (false alarm or clutter) in tracking terminology generally refers to the concept of persistency.
- A persistent false alarm (clutter) is considered a target to be tracked even if we are not interested in what or where it is.
- If one of our interesting targets gets in the vicinity of uninteresting false targets, we come prepared.

Measurement Model: targets

Target originated measurements:

$$y_t = h(x_t) + e_t, \quad \text{where } e_t \text{ is measurement noise}$$

Examples of models:

- Simple Cartesian
$$y_t = \begin{pmatrix} \mathsf{x}_t \\ \mathsf{y}_t \end{pmatrix} + e_t$$
- Bearing only
$$y_t = \operatorname{atan2}(\mathsf{y}_t, \mathsf{x}_t) + e_t$$
- Range
$$y_t = \sqrt{\mathsf{x}_t^2 + \mathsf{y}_t^2} + e_t$$
- Log range (received signal strength (RSS))
$$y_t = P_0 - \alpha \log(\mathsf{x}_t^2 + \mathsf{y}_t^2) + e_t$$

No sensor is perfect!

Measurement Model: properties

- The measurement model provides information about:
  1. the detection probability, P_D;
  2. the measured value, y_t.
- Probability of detection:
  - $P_D < 1$ in many sensors, imperfect sensors.
  - Detection probability P_D can be a characteristics of the sensor/algorithm as well as the target state. P_D *might depend on the specific target position and it can vary from target to target.*
  - It is generally difficult to find an exact formula for P_D, approximations and heuristics are needed.
- Sensor measurement noise, e_t.

Example of Sensor Model: radar (1/2)

The *radar* sensor is probably the most used sensor for ATC and target tracking applications. Today, the automotive industry is a driving force. A common measurement relation:

$$y = h(x) + e = \begin{pmatrix} \varphi \\ \theta \\ r \\ \dot{r} \end{pmatrix} + e = \begin{pmatrix} \operatorname{atan2}(\mathsf{y}/\mathsf{x}) \\ \operatorname{atan2}(\mathsf{z}/\sqrt{\mathsf{x}^2+\mathsf{y}^2}) \\ \sqrt{\mathsf{x}^2+\mathsf{y}^2+\mathsf{z}^2} \\ \frac{\mathsf{x}v^{\mathsf{x}}+\mathsf{y}v^{\mathsf{y}}+\mathsf{z}v^{\mathsf{z}}}{\sqrt{\mathsf{x}^2+\mathsf{y}^2+\mathsf{y}^2}} \end{pmatrix} + e$$

where φ is the azimuth angle, θ is the elevation, r is the range and $\dot{r}$ is the range rate (derived from the Doppler shift).

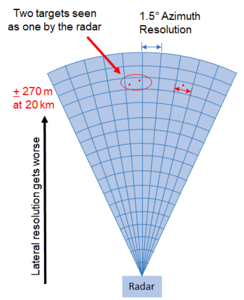
The *radar equation* also gives:

$$\text{SNR} \propto \frac{\sigma_{\text{rcs}}}{r^4},$$

where σ_{rcs} denotes the *radar cross section* (RCS). Different statistical assumptions (Swerling-cases) are used to model its PDF.

Example of Sensor Model: radar (2/2)

Radar



Radar sensor modeling, see for instance
MATLAB Sensor fusion and Tracking toolbox.

Radar modeling and techniques

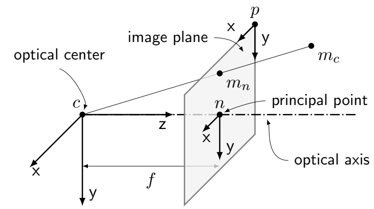
- Scan rate
- Resolution (azimuth, range)
- Accuracy: azimuth often quite accurate, but elevation not, Doppler (velocity) is very accurate
- CFAR (constant false alarm ratio)
- Techniques: Pulse radar, FMCW
- Active or passive (RWR)

Example Sensor Model: camera

- The relation between pixels coordinates (p frame) and normalized image coordinates (n frame) is given by standard calibration methods. Hence, usually, $y = m_n = \left(\begin{smallmatrix} \mathsf{x}_n \\ \mathsf{y}_n \end{smallmatrix}\right)$.
- Cameras are often modeled using the simple pin-hole camera model.
- To relate the object position to the measurement, project the point in the world, $m_c = \left(\begin{smallmatrix} x_c \\ y_c \\ z_c \end{smallmatrix}\right)$, onto the image plane to get m_n,

$$h(x) = m_n = \begin{pmatrix} \mathsf{x}_n \\ \mathsf{y}_n \end{pmatrix} = \frac{f}{\mathsf{z}_c} \begin{pmatrix} \mathsf{x}_c \\ \mathsf{y}_c \end{pmatrix}.$$

Pin-hole camera model



Measurement Model: clutter

Non-persistent measurements which do not originate from a target.

- Prior information is important.
  - Clutter maps (eg, from specification or experiments).
  - Sensor (processing algorithm) characteristics
    - Sometimes provided by the manufacturer.
    - Experiments if necessary.
- The case of minimal prior info
  - Number of false alarms (FA), m_t^{FA}, in a region of volume V: Poisson distributed with clutter rate β_{FA} (FA intensity per scan).

$$P_{\text{FA}}(m_t^{\text{FA}}) = \frac{(\beta_{\text{FA}} V)^{m_t^{\text{FA}}} e^{-\beta_{\text{FA}} V}}{m_t^{\text{FA}}!}$$

    P_{D} for the clutter is included in P_{FA}.
  - Spatial FA distribution: Uniform in the tracking volume V,

$$p_{\text{FA}}(y_t) = \frac{1}{V}$$

Measurement Model: clutter (motivation for finite resolution sensors)

Consider a sensor with finite resolution of N cells, with probability of false alarm p in each cell:

$$P_{\text{FA}}(m_t^{\text{FA}}) = \binom{N}{m_t^{\text{FA}}} p^{m_t^{\text{FA}}} (1-p)^{N-m_t^{\text{FA}}}.$$

Binomial → Poisson distribution approximation

In the limit as $N \to +\infty$ and $p \ll 1$, the binomial distribution becomes a Poisson distribution,

$$\binom{N}{m} p^m (1-p)^{N-m} \to \frac{\lambda^m e^{-\lambda}}{m!},$$

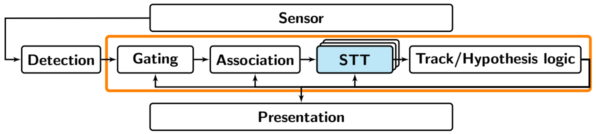
where $\lambda = Np$.

In the clutter setting, with many cells, $N \ggg 1$, and low probability of false alarm, $p \ll 1$,

$$P_{\text{FA}}(m_t^{\text{FA}}) \approx \frac{(Np)^{m_t^{\text{FA}}} e^{-Np}}{m_t^{\text{FA}}!} = \frac{(\beta_{\text{FA}} V)^{m_t^{\text{FA}}} e^{-\beta_{\text{FA}} V}}{m_t^{\text{FA}}!},$$

where Np and $\beta_{\text{FA}} V$ both represent the expected number of FA in the tracking volume.

Target Dynamics and Motion Models



LINKÖPING UNIVERSITY

Target Motion Models: constant velocity

General state-space model

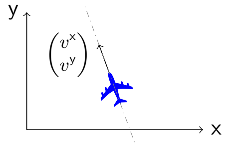
$$x_t = f(x_{t-1}) + w_t, \quad \text{where } w_t \text{ is process noise}$$

Examples of models

- (Nearly) constant velocity (CV) model

$$x_t = \begin{pmatrix} x_t \\ y_t \\ v^x_t \\ v^y_t \end{pmatrix} = \begin{pmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} x_{t-1} + \begin{pmatrix} \frac{1}{2}T^2 & 0 \\ 0 & \frac{1}{2}T^2 \\ T & 0 \\ 0 & T \end{pmatrix} a_t$$

where $a_t \sim \mathcal{N}(0, \sigma_a^2)$ is white noise.



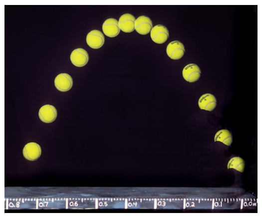
Constant velocity

Target Motion Models: constant acceleration

- (Nearly) constant acceleration (CA) model

$$x_t = \begin{pmatrix} x_t \\ v^x_t \\ a^x_t \end{pmatrix} = \begin{pmatrix} 1 & T & \frac{1}{2}T^2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{pmatrix} x_{t-1} + \begin{pmatrix} \frac{1}{2}T^2 \\ T \\ 1 \end{pmatrix} \eta_t$$

where $\eta_t \sim \mathcal{N}(0, \sigma_\eta^2)$ is white noise.



Target Motion Models: coordinated turn (1/2)

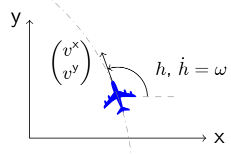
- (Nearly) coordinated turn (CT) model, *i.e.*, nearly constant speed, constant turn rate model
- State with Cartesian velocity $x_t = \begin{pmatrix} x_t & y_t & v^x_t & v^y_t & \omega_t \end{pmatrix}^T$

Continuous time description

$$\dot{x} = v\cos(h) \qquad \dot{y} = v\sin(h),$$

from which the following differential equations are obtained

$$\ddot{x} = \frac{d}{dt}\dot{x} = -v\dot{h}\sin(h) = -\omega\dot{y}$$
$$\ddot{y} = \frac{d}{dt}\dot{y} = v\dot{h}\cos(h) = \omega\dot{x}.$$

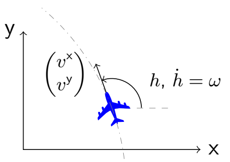


Coordinated turn

Target Motion Models: coordinated turn (2/2)

- (Nearly) coordinated turn after exact discretization

$$x_t = \begin{pmatrix} 1 & 0 & \frac{\sin(\omega_{t-1}T)}{\omega_{t-1}} & -\frac{1-\cos(\omega_{t-1}T)}{\omega_{t-1}} & 0 \\ 0 & 1 & \frac{1-\cos(\omega_{t-1}T)}{\omega_{t-1}} & \frac{\sin(\omega_{t-1}T)}{\omega_{t-1}} & 0 \\ 0 & 0 & \cos(\omega_{t-1}T) & -\sin(\omega_{t-1}T) & 0 \\ 0 & 0 & \sin(\omega_{t-1}T) & \cos(\omega_{t-1}T) & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} x_{t-1} + \begin{pmatrix} T^2/2 & 0 & 0 \\ 0 & T^2/2 & 0 \\ T & 0 & 0 \\ 0 & T & 0 \\ 0 & 0 & 1 \end{pmatrix} \eta_t$$



Coordinated turn

where $\eta_t \sim \mathcal{N}(0, \sigma_\eta^2)$ is white noise.

- The CT motion can also be defined using a polar velocity representation, which is sometimes more convenient to work with.

Continuous to Discrete Time Models (1/2)

Linear time-invariant (LTI) state-space model:

Continuous time	**Discrete time**
$\dot{x} = Ax + Bu$	$x_{t+1} = Fx_t + Gu_t$
$y = Cx + Du$	$y_t = Hx_t + Ju_t$

u is either input or process noise (then J denotes cross-correlated noise!).

Zero-order hold (ZOH) sampling

Assuming the input is piece-wise constant (ZOH):

$$\begin{aligned} x(t+T) &= e^{AT}x(t) + \int_0^T e^{A\tau} Bu(t+T-\tau)\,d\tau \\ &= \underbrace{e^{AT}}_{F} x(t) + \underbrace{\int_0^T e^{A\tau}\,d\tau\,B}_{G} u(t). \end{aligned}$$

Continuous to Discrete Time Models (2/2)

Nonlinear state-space model (two options):

Continuous time	**Discrete time**
$\dot{x} = a(x, u)$	$x_{t+1} = f(x_t, u_t)$
$y = c(x, u)$	$y_t = h(x_t, u_t)$

1. Discretized linearization (general):
   a. Linearize:
   $$A = \nabla_x a(x,u) \qquad B = \nabla_u a(x,u) \qquad C = \nabla_x c(x,u) \qquad D = \nabla_u c(x,u)$$
   b. Discretize (sample): $F = e^{AT}$, $G = \int_0^T e^{A\tau}\,d\tau\,B$, $H = C$, and $J = D$
2. Linearized discretization (best, if possible!):
   a. Discretize (sample nonlinear):
   $$x(t+T) = f\big(x(t), u(t)\big) = x(t) + \int_t^{t+T} a\big(x(\tau), u(\tau)\big)\,d\tau$$
   b. Linearize: $F = \nabla_x f(x_t, u_t)$ and $G = \nabla_u f(x_t, u_t)$

Singer Acceleration Model

Consider position, velocity and acceleration as states. Assume some memory in the acceleration:

$$\frac{d}{dt}\ddot{X}(t) = -\alpha\ddot{X}(t) + w(t),$$

where $w(t)$ is the driving white noise.

$$\dot{x}(t) = \underbrace{\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -\alpha \end{pmatrix}}_{A} x(t) + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} w(t).$$

Discretizing the system matrix assuming sample time T yields,

$$F = e^{AT} = \begin{pmatrix} 1 & T & \frac{1}{\alpha^2}(e^{-\alpha T} - 1 + \alpha T) \\ 0 & 1 & \frac{1}{\alpha}(1 - e^{-\alpha T}) \\ 0 & 0 & e^{-\alpha T} \end{pmatrix} \to \begin{pmatrix} 1 & T & \frac{T^2}{2} \\ 0 & 1 & T \\ 0 & 0 & 1 \end{pmatrix}, \text{when } \alpha T \to 0,$$

which is the constant acceleration model.

Process Noise Modeling

There are many ways to model discrete process noise given a continous model. See SF-course for more examples. Here we focus on:

$$\dot{x}(t) = a\big(x(t)\big) + w(t), \qquad \mathrm{cov}(w(t)) = \tilde{Q},$$
$$x_{t+T} = f(x_t) + w_t, \qquad \mathrm{cov}(w_t) = Q.$$

Let $f_x = \nabla_x f(x)|_{x=\hat{x}}$.

These methods correspond to more or less *ad hoc* assumptions on the process noise:

- $w(t)$ is white noise whose total influence during one sample interval,
$$Q = T\tilde{Q}.$$
- $w(t)$ is a discrete white noise sequence with variance $T\tilde{Q}$. All maneuvers occur immediately after a sample time, $x_{t+1} = f(x_t + w_t)$,
$$Q = T f_x \tilde{Q} f_x^T.$$

Models Combining Several Behaviors

Jump Markov State-Space Model (JMSSM)

$$x_t = f(x_{t-1}, \delta_t) + w_t(\delta_t)$$
$$y_t = h(x_t, \delta_t) + e_t(\delta_t)$$
$$\delta_t | \delta_{t-1} \sim p(\delta_t | \delta_{t-1})$$

where δ_t is a discrete valued Markov process, typically given by the transition matrix Π ($\Pi^{\delta_{t-1}\delta_t} = \Pr(\delta_t|\delta_{t-1})$), to indicate the current mode of the model/target.

- A target has well-defined modes.
- A target exhibit different types of behavior; *e.g.*, mixing no maneuvers and agile maneuvers

Mode Notation: comparison with tracking literature

In the tracking litterature the following notation is common for the Markovian transition model:

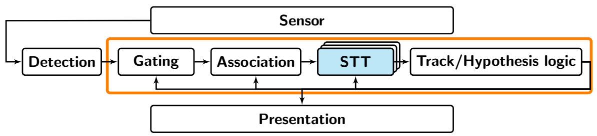
$$\Pr(M_t = M^j | M_{t-1} = M^i) = \Pr(M_t^j | M_{t-1}^i), \quad i, j = 1, \ldots, N.$$

where the mode probability and the transition probabilities are

$$\omega_t^{(j)} = \Pr(M_t^j | \mathbb{Y}_t)$$
$$\Pi^{ij} = \Pr(M_t^j | M_{t-1}^i)$$

We will use δ_t and δ_{t-1} to represent the modes, as this puts more emphasis on the associated time. Note that $\delta_t = 1, \ldots, N$ etc. if N modes are assumed.

Filter Banks

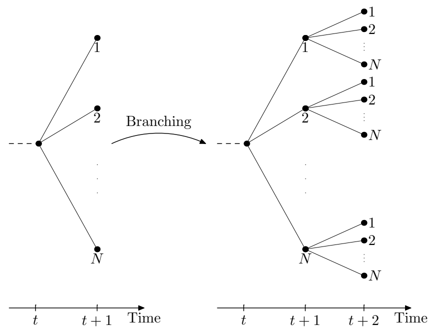


Filter Bank

- With *jump Markov state-space model* (JMSSM), both the state x_t and the mode δ_t must be considered.
- Conditioned on the mode sequence

$$\delta_{1:t} = (\delta_1, \delta_2, \dots, \delta_t),$$

  the estimate is given by an STT.
- A filter bank is an estimator with an STT for each "interesting" mode sequence, with matching probability, $\omega_{t|t}^{(\delta_{1:t})}$.
- The resulting posterior is a weighted sum of all filters in the filter bank.



Filter Bank: details

- Equations to update the filter probabilities/weights

$$\omega_{t|t-1}^{(\delta_{1:t})} = p(\delta_t|\delta_{t-1})\omega_{t-1|t-1}^{(\delta_{1:t-1})}$$

$$\omega_{t|t}^{(\delta_{1:t})} = \frac{p(y_t|\delta_{1:t}, \mathbb{Y}_{t-1})\omega_{t|t-1}^{(\delta_{1:t})}}{\sum_{\delta_{1:t}} p(y_t|\delta_{1:t}, \mathbb{Y}_{t-1})\omega_{t|t-1}^{(\delta_{1:t})}}$$

- Resulting posterior distribution

$$p(x_t|\mathbb{Y}_t) = \sum\nolimits_{\delta_{1:t}} \omega_{t|t}^{(\delta_{1:t})} p(x_t|\mathbb{Y}_t, \delta_{1:t})$$

- The MMSE given STT estimates with mean and covariance $(\hat{x}^{(\delta)}, P^{(\delta)})$ become:

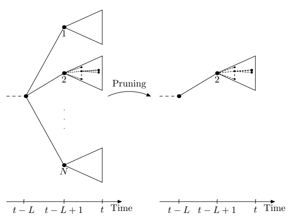
$$\hat{x} = \sum\nolimits_{\delta} \omega^{(\delta)} \hat{x}^{(\delta)}$$

$$P = \sum\nolimits_{\delta} \omega^{(\delta)} \big(P^{(\delta)} + \underbrace{(\hat{x}^{(\delta)} - \hat{x})(\hat{x}^{(\delta)} - \hat{x})^T}_{\text{Spread of the mean}}\big).$$

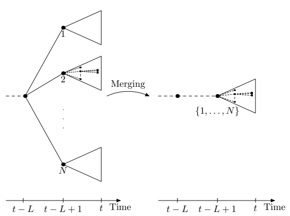
Filter Bank: problem

- Filter banks grows with combinatorial complexity, hence it quickly becomes unmanageable.
- Common approximations:

  Pruning: Drop unlikely branches.
  Merging: Combine branches with recent common heritage.

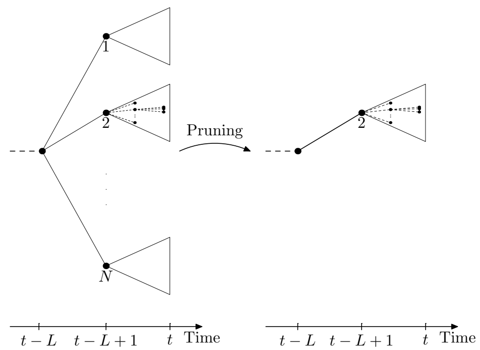


Pruning



Merging

Filter Bank Approximation: pruning



- Prune branches with low probability:
  - Mode sequences with too low probability.
  - "Trees" with too low accumulated probability since L steps back.
- After reducing the filter bank to suitable size, re-normalize the remaining weights, $\delta \in \Delta$, such that

$$\sum_{\delta \in \Delta} \omega^{(\delta)} = 1.$$

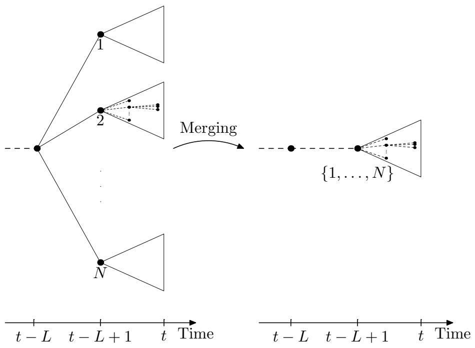
Filter Bank Approximation: merging

- Reduce the filter bank by combining mode sequences that have recently been similar.
- The weight of the merged mode sequences, $\delta \in \Delta$, are add up to the weight of the merged branch, δ',

$$\omega^{(\delta')} = \sum\nolimits_{\delta \in \Delta} \omega^{(\delta)}.$$

- The mean and covariance become

$$\hat{x}^{(\delta')} = \tfrac{1}{\omega^{(\delta')}} \sum\nolimits_{\delta \in \Delta} \omega^{(\delta)} \hat{x}^{(\delta)}$$
$$P^{(\delta')} = \tfrac{1}{\omega^{(\delta')}} \sum\nolimits_{\delta \in \Delta} \omega^{(\delta)} \big(P^{(\delta)} + (\hat{x}^{(\delta)} - \hat{x})(\hat{x}^{(\delta)} - \hat{x})^T\big).$$



Filter Banks in Target Tracking

- Use different models to capture mixed behaviors, with different modes to bring out the most of the measurements.
- Maneuvers can be quickly detected (or can be ignored), hence a shallow tree is enough.
- Common tracking modes:
  - No maneuver (CV model)
  - Medium maneuvers (CA model)
  - Turns (CT model)

Examples of algorithms

- *Generalized pseudo Bayesian* of depth n (GPB(n)) filter
- *Interacting multiple models* (IMM) filter

Preliminaries: probability theory

Reminder

- Conditional probability: $\Pr(A|B) = \frac{\Pr(A,B)}{\Pr(B)}$
- Note that we can condition: $\Pr(A|B,C) = \frac{\Pr(A,B|C)}{\Pr(B|C)}$
- Bayes' rule: $\Pr(A|B) = \frac{\Pr(B|A)\Pr(A)}{\Pr(B)}$
- Note: Common quantities for A, B and C: x_t and $\mathbb{Y}_t = \{y_t, \mathbb{Y}_{t-1}\}$
- Total probability theorem: $\Pr(A) = \sum_\delta \Pr(A|\delta)\Pr(\delta)$

Example: $A = x_t$, $B = y_t$, **and** $C = \mathbb{Y}_{t-1}$

$$p(x_t|\mathbb{Y}_t) = p(x_t|y_t, \mathbb{Y}_{t-1}) = \frac{p(y_t|x_t, \mathbb{Y}_{t-1})p(x_t|\mathbb{Y}_{t-1})}{p(y_t|\mathbb{Y}_{t-1})}$$

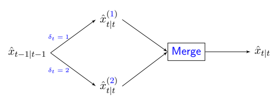
Common Approximations: GPB and IMM filter

To avoid exponential complexity for the multiple model approach, methods have been developed using the above approximation.

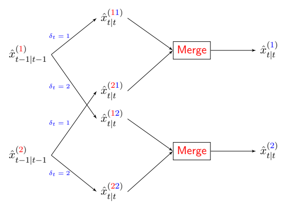
Assuming r different parallel models:

- GPB(1): Merge the filter bank at the current level. Complexity: $\mathcal{O}(r)$
- GPB(2): Merge the filter bank at one step back. Complexity: $\mathcal{O}(r^2)$
- IMM: A more efficient implementation of GPB(2). Complexity: $\mathcal{O}(r)$

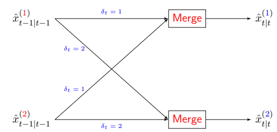
GPB(1)



GPB(2)



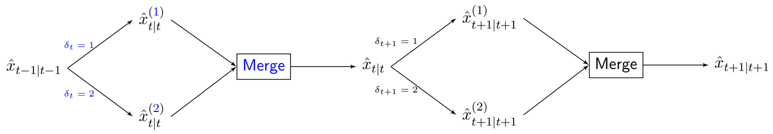
IMM



GPB Filtering: illustration

Generalized Psuedo Bayesian (GPB(1)) filtering

KF-filter bank hypotheses are merged to a single mode after each measurement update.



GPB(1) with $r = 2$ models

Multiple Models: GPB(1) derivation (1/2)

Assume the following prior:

$$p(x_{t-1}|\mathbb{Y}_{t-1}) = \mathcal{N}(x_{t-1}; \hat{x}_{t-1|t-1}, P_{t-1|t-1})$$

Then the posterior can be computed according to

$$\begin{aligned} p(x_t|\mathbb{Y}_t) &= \sum_{\delta_t} p(x_t|\delta_t, \mathbb{Y}_t) \Pr(\delta_t|\mathbb{Y}_t) \\ &= \sum_{\delta_t} \omega_t^{(\delta_t)} p(x_t|\delta_t, y_t, \hat{x}_{t-1|t-1}, P_{t-1|t-1}) \\ &\approx \sum_{\delta_t} \omega_t^{(\delta_t)} \mathcal{N}(x_t; \hat{x}^{(\delta_t)}_{t|t}, P^{(\delta_t)}_{t|t}) \\ &\approx \mathcal{N}(x_t; \hat{x}_{t|t}, P_{t|t}) \end{aligned}$$

Multiple Models: GPB(1) derivation (2/2)

Mode likelihood computation:

Each mode contribute $\hat{x}^{(\delta_t)}_{t|t}$ and $P^{(\delta_t)}_{t|t}$ to the final estimate $\hat{x}_{t|t}$ based on their likelihood,

$$\begin{aligned} \omega_t^{(\delta_t)} &= \Pr(\delta_t|\mathbb{Y}_t) = \Pr(\delta_t|y_t, \mathbb{Y}_{t-1}) \propto p(y_t|\delta_t, \mathbb{Y}_{t-1}) \Pr(\delta_t|\mathbb{Y}_{t-1}) \\ &= p(y_t|\delta_t, \mathbb{Y}_{t-1}) \Pr(\delta_t) = p(y_t|\delta_t, \mathbb{Y}_{t-1}) \Pi^{\delta_t}, \end{aligned}$$

where Π^{δ_t} is the probability to end up in mode δ_t at time t, which is a simplified form of the transition matrix Π given that we have marginalized away the complete mode history.

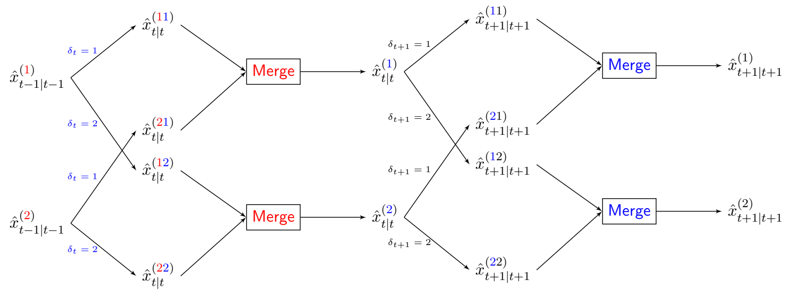
Mode reduction using merging:

$$\hat{x}_{t|t} = \sum_{\delta_t} \omega^{(\delta_t)} \hat{x}^{(\delta_t)}_{t|t}, \quad \text{and} \quad P_{t|t} = \sum_{\delta_t} \omega^{(\delta_t)} (P^{(\delta_t)}_{t|t} + (\hat{x}^{(\delta_t)}_{t|t} - \hat{x}_{t|t})(\hat{x}^{(\delta_t)}_{t|t} - \hat{x}_{t|t})^T)$$

GPB Filtering: illustration

Generalized Psuedo Bayesian (GPB(2)) filtering

KF-filter bank hypotheses merged to a depth after each measurement update.



GPB(2) with $r = 2$ models

Multiple Models: GPB(2) derivation (1/3)

Assume the following prior:

$$p(x_{t-1}|\mathbb{Y}_{t-1}) = \sum_{\delta_{t-1}} \omega_{t-1}^{(\delta_{t-1})} \mathcal{N}(x_{t-1}; \hat{x}_{t-1|t-1}^{(\delta_{t-1})}, P_{t-1|t-1}^{(\delta_{t-1})})$$

Then the posterior can be computed according to

$$p(x_t|\mathbb{Y}_t) = \sum_{\delta_t, \delta_{t-1}} \color{blue}{p(x_t|\delta_t, \delta_{t-1}, \mathbb{Y}_t)} \color{red}{p(\delta_{t-1}|\delta_t, \mathbb{Y}_t)} \color{green}{p(\delta_t|\mathbb{Y}_t)}$$

Where the first term is the filter estimate assuming the mode sequence $\delta_{t-1}\delta_t$:

$$\color{blue}{p(x_t|\delta_t, \delta_{t-1}, \mathbb{Y}_t)} = \mathcal{N}(x_t; \hat{x}_{t|t}^{(\delta_{t-1}\delta_t)}, P_{t|t}^{(\delta_{t-1}\delta_t)})$$

Multiple Models: GPB(2) derivation (2/3)

The two remaining terms are:

$$\begin{aligned}
\color{red}{p(\delta_{t-1}|\delta_t, \mathbb{Y}_t)} &= \frac{p(y_t|\delta_{t-1}, \delta_t, \mathbb{Y}_{t-1}) p(\delta_{t-1}|\delta_t, \mathbb{Y}_{t-1})}{p(y_t|\delta_t, \mathbb{Y}_{t-1})} \\
&= \frac{p(y_t|\delta_{t-1}, \delta_t, \mathbb{Y}_{t-1}) p(\delta_t|\delta_{t-1}, \mathbb{Y}_{t-1}) p(\delta_{t-1}|\mathbb{Y}_{t-1})}{p(y_t|\delta_t, \mathbb{Y}_{t-1}) p(\delta_t|\mathbb{Y}_{t-1})} \\
&= \frac{p(y_t|\delta_{t-1}, \delta_t, \mathbb{Y}_{t-1}) \Pi^{\delta_{t-1}\delta_t} \omega_{t-1}^{(\delta_{t-1})}}{p(y_t, \delta_t|\mathbb{Y}_{t-1})} \\
\color{green}{p(\delta_t|\mathbb{Y}_t)} &= \frac{p(y_t, \delta_t|\mathbb{Y}_{t-1})}{p(y_t|\mathbb{Y}_{t-1})}
\end{aligned}$$

Note the two terms canceling when the two terms are multiplied.

Multiple Models: GPB(2) derivation (3/3)

Putting it all together

$$p(x_t|\mathbb{Y}_t) \propto \sum_{\delta_{t-1}, \delta_t} \underbrace{p(y_t|\delta_{t-1}, \delta_t, \mathbb{Y}_{t-1}) \Pi^{\delta_{t-1}\delta_t} \omega_{t-1}^{(\delta_{t-1})}}_{\propto \omega_t^{(\delta_{t-1}\delta_t)}, \quad \sum \omega_t^{(\delta_{t-1}\delta_t)} = 1} \mathcal{N}(x_t; \hat{x}_{t|t}^{(\delta_{t-1}\delta_t)}, P_{t|t}^{(\delta_{t-1}\delta_t)})$$

$$p(x_t|\mathbb{Y}_t) \approx \sum_{\delta_t} \omega_t^{(\delta_t)} \mathcal{N}(x_t; \hat{x}_{t|t}^{(\delta_t)}, P_{t|t}^{(\delta_t)})$$

$$\omega_t^{(\delta_t)} = \sum_{\delta_{t-1}} \omega_t^{(\delta_{t-1}\delta_t)}$$

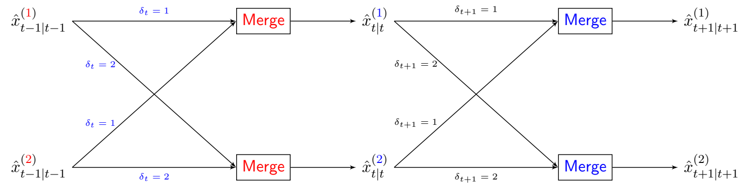
$$\hat{x}_{t|t}^{(\delta_t)} = \frac{1}{\omega_t^{(\delta_t)}} \sum_{\delta_{t-1}} \omega_t^{(\delta_{t-1}\delta_t)} \hat{x}_{t|t}^{(\delta_{t-1}\delta_t)}$$

$$P_{t|t}^{(\delta_t)} = \frac{1}{\omega_t^{(\delta_t)}} \sum_{\delta_{t-1}} \omega^{(\delta_{t-1}\delta_t)} (P_{t|t}^{(\delta_{t-1}\delta_t)} + (\hat{x}_{t|t}^{(\delta_{t-1}\delta_t)} - \hat{x}_{t|t}^{(\delta_t)})(\hat{x}_{t|t}^{(\delta_{t-1}\delta_t)} - \hat{x}_{t|t}^{(\delta_t)})^T)$$

IMM Filtering

Interacting multiple models (IMM) filtering

IMM is an alternative implementation of the GPB(2), which achieves lower computational complexity using a clever reordering of the computations. It has become a standard solution.



IMM filter with 2 models

Multiple Models: derivations IMM (1/2)

Total probability theorem:

$$p(x_t|\mathbb{Y}_t) = \sum_{\delta_t} p(x_t|\delta_t, \mathbb{Y}_t)p(\delta_t|\mathbb{Y}_t) = \sum_{\delta_t} p(x_t|\delta_t, y_t, \mathbb{Y}_{t-1})\omega_t^{(\delta_t)}$$

Baye's rule:

$$p(x_t|\delta_t, y_t, \mathbb{Y}_{t-1}) = \frac{p(y_t|\delta_t, x_t)p(x_t|\delta_t, \mathbb{Y}_{t-1})}{p(y_t|\delta_t, \mathbb{Y}_{t-1})}$$

Multiple Models: derivations IMM (2/2)

Total probability theorem now gives:

$$\begin{aligned}
p(x_t|\delta_t, \mathbb{Y}_{t-1}) &= \sum_{\delta_{t-1}} p(x_t|\delta_{t-1}, \delta_t, \mathbb{Y}_{t-1}) \underbrace{p(\delta_{t-1}|\delta_t, \mathbb{Y}_{t-1})}_{\mu_{t-1}^{\delta_{t-1}|\delta_t}} \\
&\approx \sum_{\delta_{t-1}} \mu_{t-1}^{\delta_{t-1}|\delta_t} p(x_t|\delta_t, \delta_{t-1}, \hat{x}_{t-1|t-1}^{(\delta_{t-1})}, P_{t-1|t-1}^{(\delta_{t-1})}) \\
&= \sum_{\delta_{t-1}} \mu_{t-1}^{\delta_{t-1}|\delta_t} \mathcal{N}(x_t; \mathsf{E}(x_t|\delta_t, \hat{x}_{t-1|t-1}^{(\delta_{t-1})}), \mathsf{cov}(x_t|\delta_t, \hat{x}_{t-1|t-1}^{(\delta_{t-1})})) \\
&\approx \mathcal{N}(x_t; \sum_{\delta_{t-1}} \mu_{t-1}^{\delta_{t-1}|\delta_t} \, \mathsf{E}(x_t|\delta_t, \hat{x}_{t-1|t-1}^{(\delta_{t-1})}), \mathsf{cov}(\star)) \\
&= \mathcal{N}(x_t; \mathsf{E}(x_t|\delta_t, \sum_{\delta_{t-1}} \mu_{t-1}^{\delta_{t-1}|\delta_t} \hat{x}_{t-1|t-1}^{(\delta_{t-1})}), \mathsf{cov}(\star))
\end{aligned}$$

Multiple Models: IMM algorithm (1/2)

- Calculate **mixing probabilities:**

$$\mu_{t-1}^{\delta_{t-1}|\delta_t} \propto \Pi^{\delta_{t-1}\delta_t}\omega_{t-1}^{(\delta_{t-1})}, \quad \sum_{\delta_{t-1}} \mu_{t-1}^{\delta_{t-1}|\delta_t} = 1$$

- **Mixing:** Start with $\hat{x}_{t-1|t-1}^{(\delta_{t-1})}$ and $P_{t-1|t-1}^{(\delta_{t-1})}$.

$$\hat{x}_{t-1|t-1}^{(0\delta_t)} = \sum_{\delta_{t-1}} \mu_{t-1}^{\delta_{t-1}|\delta_t} \hat{x}_{t-1|t-1}^{(\delta_{t-1})}$$

$$P_{t-1|t-1}^{(0\delta_t)} = \sum_{\delta_{t-1}} \mu_{t-1}^{\delta_{t-1}|\delta_t} (P_{t-1|t-1}^{(\delta_{t-1})} + (\hat{x}_{t-1|t-1}^{(\delta_{t-1})} - \hat{x}_{t-1|t-1}^{(0\delta_t)})(\hat{x}_{t-1|t-1}^{(\delta_{t-1})} - \hat{x}_{t-1|t-1}^{(0\delta_t)})^T)$$

Multiple Models: IMM algorithm (2/2)

- **Mode-matched filtering:**

$$\Lambda_t^{(\delta_t)} = p(y_t|\delta_t, \hat{x}_{t-1|t-1}^{(0\delta_t)}, P_{t-1|t-1}^{(0\delta_t)}).$$

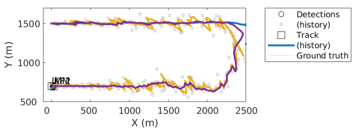
  Update $(\hat{x}_{t-1|t-1}^{(0\delta_t)}, P_{t-1|t-1}^{(0\delta_t)})$ with the measurement y_t to obtain the new filter modes $(\hat{x}_{t|t}^{(\delta_t)}, P_{t|t}^{(\delta_t)})$.

- **Mode probability update:**

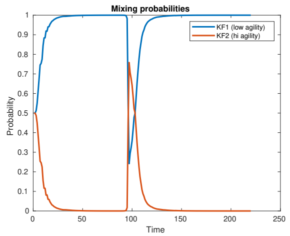
$$\omega_t^{(\delta_t)} \propto \Lambda_t^{(\delta_t)} \sum_{\delta_{t-1}} \Pi^{\delta_{t-1}\delta_t}\omega_{t-1}^{(\delta_{t-1})}$$

IMM Filter Illustration I

A radar tracking application is presented using a two filter IMM filter. One filter is used to handle a straight paths, whereas the other is used to manage maneuvers. Due to the nonlinearities in the measurement equation an EKF is used for the estimation.

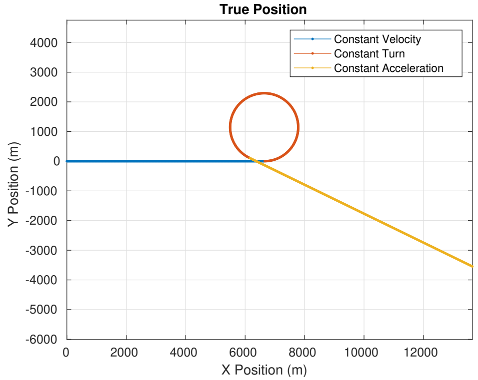


https://youtu.be/DVkCzdku2SQ



IMM Filter Illustration II (1/3)

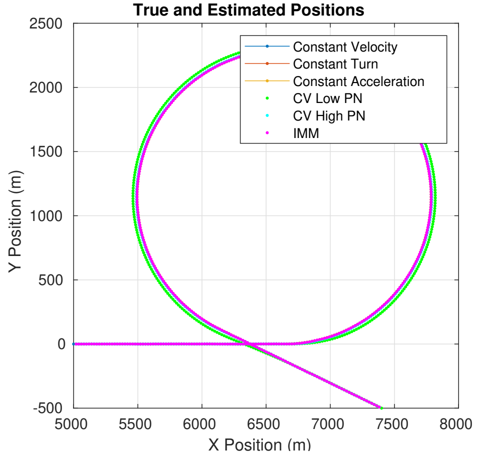
- Simulated trajectory with CV, CT, and CA segments
- Position measurements
- Compared filters:
  - KF with CV low process noise
  - KF with CV high process noise
  - IMM filter with CV, CT, and CA models



Simulated trajectory

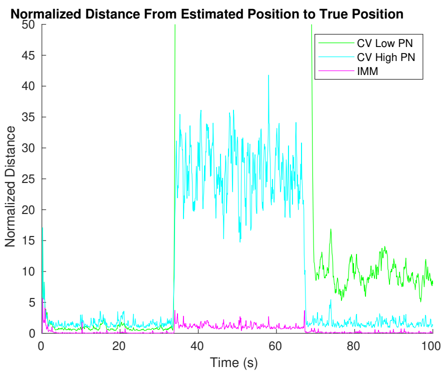
Example taken from MATLAB Sensor Fusion and Tracking toolbox.

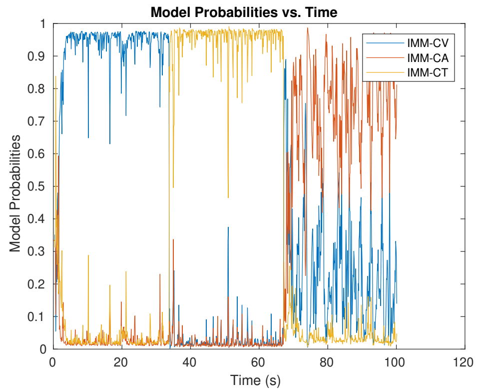
IMM Filter Illustration II (2/3)



- The low process noise KF clearly cannot keep up.
- The high process noise KF, keeps up better but is slightly noisier than the IMM filter.
- Differences not very visible in this plot.
- The predominant models in the IMM matches the simulated trajectory well.

IMM Filter Illustration II (3/3)

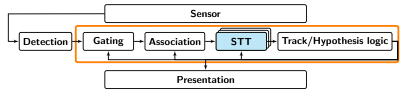




Summary

LINKÖPING UNIVERSITY

Summary



- Common tracking sensors: range, bearing, range-rate (Doppler shift), ...
  Models are derived from physical relations, and assumptions about noise levels.
- Common motion target models: constant velocity, constant acceleration, coordinated turn
  Coarse approximations to allow for reasonable target maneuvers, are derived from basic physical relations
- Mixtures are a common tool to cover several different possible behaviors
  Maneuvering targets are commonly tracked using IMM filters, which approximate the complete filter bank solution