Bayesian Filters for Location Estimation and Tracking – An Introduction

Traian E. Abrudan

tabrudan@fe.up.pt

Instituto de Telecomunicações, Departamento de Engenharia Electrotécnica e Computadores, Universidade do Porto, Portugal
Outline

• Motivation
• Deterministic vs. probabilistic models
• Probabilistic models for location estimation and tracking
• Bayesian filters
  – Kalman filter and its extensions
  – Grid-based filter
  – Particle filter
  – Pros. and cons.
• Real-world results
• Conclusions
Motivation

• Indoor propagation environment is extremely harsh
• Signals experience deep fades in space, time and frequency
• As a result of all these effects, the propagation channel behaves randomly
• Measurements are subject to random errors due to
  – random noise at the sensors
  – device manufacturing imperfections
  – imperfect calibration
• Dealing with uncertainties calls for probabilistic models
• Powerful statistical estimators may be derived
• They do not just provide an estimated value, they also provide reliability information
Deterministic vs. statistical models

• Deterministic models for location estimation are quite “rough”
  – perform “hard decisions” (quantize the estimated parameters)
  – discard valuable statistical information embedded in the data

• Probabilistic models exploit the available statistical information
  – Parameters are modeled as random variables with the corresponding *probability density functions (p.d.f.’s)*
  – Prior knowledge on the errors (e.g. from the measurements) may be included in the model in order to improve the parameter estimation process
  – Optimality can be achieved (w.r.t the assumed prior)
Probabilistic models
for location estimation and tracking

• Let $x_k$ denote the $L_S \times 1$ state vector at time instance $k$, with $x_k \in \mathcal{D}_x \subset \mathbb{R}^{L_S}$ (or $\mathbb{C}^{L_S}$)

• $x_k$ may include the position (but also velocity, acceleration, heading, etc.)

• Let $z_k$ be the observation vector at time $k$ (e.g. RSSI, ToA, AoA, etc., or combinations of those)

• Our goal is to estimate the sequence of states $x_k$, $k = 0, 1 \ldots$, based on all available measurements up to time $k$ (abbreviated $z_{1:k}$)

• For this reason, the process is called filtering

• The corresponding posterior p.d.f. we are interested in is $p(x_k | z_{1:k})$

• Assume that the (hidden) true states $x_k$ are connected in a 1st-order Markov chain – Hidden Markov Model (HMM)
Probabilistic models for location estimation and tracking

Markov model assumptions:

- The current true state is conditionally independent of all previous states given the last state: \( p(x_k | x_{1:k-1}) = p(x_k | x_{k-1}) \)
- Observations \( z_{1:k} \) are conditionally independent provided that \( x_0, \ldots, x_k \), are known: \( p(z_k | x_{1:k}) = p(z_k | x_k) \)
Probabilistic models
for location estimation and tracking

- State-space model:

  \[
  \text{prediction equation: } \quad x_k = f(x_{k-1}, u_k) + w_k \quad \rightarrow \quad p(x_k|x_{k-1}) \tag{1}
  \]

  \[
  \text{measurement equation: } \quad z_k = h(x_k) + v_k \quad \rightarrow \quad p(x_k|z_k) \tag{2}
  \]

  \(f, h\) are known functions

  \(u_k\) is a known input (required only if a control input exists)

  \(w_k\) is the state (process) noise

  \(v_k\) is the measurement noise

  \(w_k, v_k\) are i.i.d., mutually independent, with known p.d.f.’s

- For localization
  - prediction: a motion model
  - measurement: RSSI, ToA, AoA, kinematic parameters, etc., or combinations

- **Problem:** what to trust more: *the prediction* or *the measurement*?

- The Bayesian estimator solves this problem reliably
Probabilistic models
for location estimation and tracking

• State-space model:

\[
prediction\ equation:\quad x_k = f(x_{k-1}, u_k) + w_k \quad \rightarrow \quad p(x_k|x_{k-1}) \quad (3)
\]

\[
measurement\ equation:\quad z_k = h(x_k) + v_k \quad \rightarrow \quad p(x_k|z_k) \quad (4)
\]

• We would like to derive a formula such that the new posterior p.d.f.
at time \(k\), \(p(x_k|z_{1:k})\) is obtained by updating the old posterior at
time \(k-1\), \(p(x_{k-1}|z_{1:k-1})\)

• This way, the filter can operate sequentially, in real-time (online)

• Moreover, we store only the parameters at the previous time
instance and avoid memory growth
Probabilistic models for location estimation and tracking

- Assume that the old posterior $p(x_{k-1}|z_{1:k-1})$ is available at time $k$
- **Prediction step:** $p(x_{k-1}|z_{1:k-1}) \rightarrow p(x_k|z_{1:k-1})$
- Using Chapman-Kolmogorov equation:

$$p(x_k|z_{1:k-1}) = \int_{D_x} p(x_k|x_{k-1})p(x_{k-1}|z_{1:k-1})dx_{k-1}$$

(5)

- This is the prior of the state $x_k$ *without knowing* the incoming measurement $z_k$, knowing only the previous measurements $z_{1:k-1}$
- The prediction step usually deforms / translates / spreads the p.d.f. due to noise
Probabilistic models for location estimation and tracking

- **Update step:** \( \left\{ p(x_k|z_{1:k-1}), z_k \right\} \rightarrow p(x_k|z_{1:k}) \)

\[
p(x_k|z_{1:k}) = \frac{p(z_k|x_k)p(x_k|z_{1:k-1})}{p(z_k|z_{1:k-1})}, \quad \text{i.e., posterior} = \frac{\text{likelihood} \cdot \text{prior}}{\text{evidence}} \tag{6}
\]

- The denominator \( c = p(z_k|z_{1:k-1}) = \int_{\mathcal{X}} p(z_k|x_k)p(x_k|z_{1:k-1}) \, dx_k \) is just a normalization constant (independent of \( x_k \))

- The update combines the likelihood of the received measurement with the predicted state

- The update step usually concentrates the p.d.f.

- A closed-form expression that relates the *old posterior* to the *new posterior* is obtained by replacing eq. (5) into eq. (6)
Probabilistic models for location estimation and tracking

- Sequential update of the prior $p(x_{k-1}|z_{1:k-1})$, $z_k \to p(x_k|z_{1:k})$:

$$p(x_k|z_{1:k}) = \frac{1}{c} \int_{D_x} p(z_k|x_k) p(x_k|x_{k-1}) \int_{D_x} p(x_{k-1}|z_{1:k-1}) \, dx_{k-1}$$ (7)

- This theoretically allows an optimal Bayesian solution – Minimum Mean Square Error (MMSE), Maximum a posteriori (MAP) estimators, etc.

- Unfortunately, this is just a conceptual solution, integrals are intractable

- In some cases (under restrictive assumptions), (close to) optimal tractable solutions are obtained:
  - Kalman filter [Sin70, ParLee01, AbrPauBar11]
  - grid-based filter [Muh03]
Kalman filter (KF)

- State-space model:

  prediction equation: \[ x_k = F_k x_{k-1} + B_k u_k + G_k w_k \] (8)
  measurement equation: \[ z_k = H_k x_k + v_k \] (9)

where \( F_k, B_k, G_k, H_k \) are known matrices

- When noises are zero-mean jointly Gaussian, Kalman filter is optimal estimator in the mean-square error (MSE) sense

- Otherwise, KF is the best linear estimator

- It finds the posterior mean \( \mathbb{E}\{x_k|z_{1:k}\} = \hat{x}_k|k \) and its covariance
  \( \mathbb{E}\{(x_k - \hat{x}_k|k)(x_k - \hat{x}_k|k)^T\} \) and updates them sequentially

  \[ \hat{x}_k|k = \hat{x}_k|k-1 - K_k [z_k - F_k \hat{x}_k|k-1] \] (10)

where \( K_k \) is the Kalman gain, and the subscript \( (\cdot)_{k|k-1} \) denotes the \textit{a priori} estimated state (before considering the measurement \( z_k \))
Kalman filter and its extensions (EKF, UKF)

• Extended Kalman filter (EKF) – an extension of KF to non-linear state-space equations
  √ either the process is non-linear, or the measurements are not a linear function of the states
  √ EKF linearizes the model about the new estimate (similar to Taylor series approximation)
  √ works well in many situations, but may diverge for highly non-linear models (covariance is propagated through linearization)

• Unscented Kalman filter (UKF) – mean and covariance are projected via the so-called *unscented transform*
  √ picks up a minimal set of sample points around the mean – called *sigma points* – propagates those through the non-linearity
  √ UKF can deal with highly-nonlinear models
  √ often, UKF works better than EKF
Kalman filter and its extensions (EKF, UKF)

• KF, EKF, UKF do not work very well for p.d.f.’s that have
  - heavy-tails / high kurtosis
• They may totally fail for
  - heavily skewed p.d.f.’s
  - bimodal/multimodal p.d.f.’s
• We need more general filters to tackle these problems
• It is everything about approximating well integrals and the p.d.f.’s in
  the optimal Bayes estimation of the posterior

\[
p(x_k | z_{1:k}) = \frac{1}{c} \ p(z_k | x_k) \int_{D_x} p(x_k | x_{k-1}) p(x_{k-1} | z_{1:k-1}) \ dx_{k-1}
\]

• The price we pay is the high complexity – curse of dimensionality
  [DauHua03]
Grid-based filter

- Limit the total number of possible states to $N_S$ [AruMas02, Muh03]
- Define a grid of discrete states $x^i$ (may also be different at each time instance)
- The p.d.f.'s are discretized as well
- The conditional probability of state $x^i$ given the measurements $z_{1:k-1}$ is

\[
P(x_{k-1} = x^i | z_{1:k-1}) = \omega^i_{k-1|k-1}
\]  

(11)

- Then, the old posterior may be written as a sum of Dirac pulses

\[
p(x_{k-1} | z_{1:k-1}) = \sum_{i=1}^{N_S} \omega^i_{k-1|k-1} \delta(x_{k-1} - x^i)
\]  

(12)
Grid-based filter

- Both the new prior and the new posterior have the same structure: a sum of weighted and delayed Dirac impulses

\[
p(x_k|z_{1:k-1}) = \sum_{i=1}^{N_S} \omega_{k|k-1}^i \delta(x_{k-1} - x^i) \tag{13}
\]

\[
p(x_k|z_{1:k}) = \sum_{i=1}^{N_S} \omega_{k|k}^i \delta(x_{k-1} - x^i) \tag{14}
\]
Grid-based filter

- The prediction step: the Chapman-Kolmogorov equation (5) becomes:

\[
p(x_k | z_{1:k-1}) = \sum_{i=1}^{N_S} \sum_{j=1}^{N_S} \omega^i_{k-1} p(x^i | x^j) \delta(x_{k-1} - x^i)
\]

(15)

\[
= \sum_{i=1}^{N_S} \omega^i_{k-1} \delta(x_{k-1} - x^i)
\]

(16)

where

\[
\omega^i_{k-1} = \sum_{j=1}^{N_S} \omega^j_{k-1} p(x^i | x^j)
\]

(17)

new prior weights = old posterior weights, reweighted using state transition probabilities
Grid-based filter

- The update step: equation (6) becomes:

\[
p(x_k | z_{1:k}) = \sum_{i=1}^{N_S} \omega_{k|k}^i \delta(x_{k-1} - x^i)
\]

where

\[
\omega_{k|k}^i = \frac{\omega_{k|k-1}^i p(z_k | x^i)}{\sum_{j=1}^{N_S} \omega_{k|k-1}^j p(z_k | x^j)}
\]

posterior weights = prior weights, reweighted using likelihoods

- Grid-based filters are computationally expensive
- Grid resolution needs to be sufficiently high
Particle filter

- They are sophisticated model estimation techniques based on simulation
- Also known as Sequential Monte Carlo (SMC) methods [GusGun02, AruMas02, Muh03]
- Integrals involved in the Bayesian filter cannot be solved analytically
- The idea: represent the p.d.f.’s by using a set of random “particles” (random = Monte Carlo method)
- Particles are just differently weighted samples of the distribution
- By increasing the number of samples, the representation almost sure converges to the true p.d.f.
Particle filter

Figure 2: Representation of the Gaussian p.d.f. $\mathcal{N}(1, 1)$ using particles (1-D case). The particle weight is represented by its size.
Particle filter

2-D Gaussian p.d.f.

equal weight particles

non–equal weight particles

Figure 3: Representation of the Gaussian p.d.f. using particles (1-D case). The particle weight is represented by its size. The p.d.f. is $\mathcal{N}([0.5; 1], [1.6 \ 0.8; 0.8 \ 0.8])$
Particle filter  
Sequential Importance Sampling (SIS)

- SIS is the basic framework for most particle filter algorithms
- Let \( \{x_{0:k}^i\} \) be a set of points (particles), \( i = 1, \ldots, N_P \)
- This describes the whole trajectory in the state space for each particle \( i \)
- Let \( \omega_k^i \) be the associated weights, normalized such that \( \sum_{i=1}^{N_P} \omega_k^i = 1 \)
- Then, the discrete weighted approximation of the posterior p.d.f. is

\[
p(x_k|z_{1:k}) \approx \sum_{i=1}^{N_P} \omega_k^i \delta(x_{0:k} - x_{0:k}^i) \tag{20}
\]
Particle filter
Sequential Importance Sampling (SIS)

- Usually, we cannot draw samples $x_k^i$ from $p(\cdot)$ directly
- Assume we sample directly for a (different) importance density $q(\cdot)$
- Our approximation is still correct (up to normalization) if

$$\omega_k^i \propto \frac{p(x_{0:k}^i|z_{1:k})}{q(x_{0:k}^i|z_{1:k})}$$  \hspace{1cm} (21)

- The benefit is that we can choose $q(\cdot)$ freely
- If the importance function is chosen to factorize such that

$$q(x_{0:k}|z_{1:k}) = q(x_k|x_{0:k-1}, z_{1:k})q(x_{0:k-1}|z_{1:k-1})$$  \hspace{1cm} (22)

then one can augment the old particles $x_{0:k-1}^i$ by

$$x_k \sim q(x_k|x_{0:k-1}, z_{1:k})$$ to get the new particles $x_{0:k}^i$.
Particle filter
Sequential Importance Sampling (SIS)

• The weight update is of form

\[ \omega_k^i = \omega_{k-1}^i \frac{p(z_k^i|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{0:k-1}, z_{1:k})} \] (23)

• Furthermore, if \( q(x_k^i|x_{0:k-1}, z_{1:k}) = q(x_k^i|x_{k-1}, z_{1:k}) \) (only dependent on the last state), then we do not need to preserve the full trajectory of the particles \( x_{0:k-1} \), neither the observations \( z_{1:k} \)
Particle filter
Degeneracy Problem

• The problem with SIS approach is that after few iterations, most particles have negligible weight

• The weight is concentrated on only few particles – the one that closely match the measurements

• Solutions:
- brute force: huge number of particles – too complex
- good choice of importance density
- resampling
Particle filter
Optimal importance density

• It can be shown that the optimal importance density is given by

\[ q_{\text{opt}}(x_k|x_{k-1}, z_k) = p(x_k|x_{k-1}, z_k) \quad (24) \]

• Then

\[ \omega_k^i = \omega_{k-1}^i \int_{D_x} p(z_k|x_k')p(x_k'|x_{k-1}^i)dx_k' \quad (25) \]

• Drawing sample from the optimal distribution is not always possible

• One alternative is to choose the importance density to be the prior

\[ q(x_k|x_{k-1}, z_k) = p(x_k|x_{k-1}) \quad (26) \]

• This is simple, but does not take into account the measurements
Particle filter

Resampling

• Basic idea of resampling:

Whenever the degeneracy exceeds some threshold, replace the old samples (+ weights) with a new set of samples (+ weights), such that the sample density reflects better the posterior p.d.f.

• This eliminates particles with low weights and chooses new particles in the more probable regions

• Complexity is possible in $O(N_p)$
Figure 4: Illustration of the resampling process.
Particle filter
Pseudo-code [Muh03, AruMas02]

\[
\{x^i_k, \omega^i_k\}_{i=1}^{N_P} = \text{PF}(\{x^i_{k-1}, \omega^i_{k-1}\}_{i=1}^{N_P}, z^i_k)
\]

\[
\text{FOR } i = 1 \text{ to } N_P
\]
\[
draw x^i_k \sim q(x_k | x^i_{k-1}, z_k)
\]
update weights according to

\[
\omega^i_k = \omega^i_{k-1} \frac{p(z_k | x^i_k)p(x^i_k | x^i_{k-1})}{q(x^i_k | x_{0:k-1}, z_{1:k})}
\]

\[
\text{END FOR}
\]

normalize weights to \(\sum_{i=1}^{N_P} \omega^i_k = 1\)

IF degeneracy is too high
resample \(\{x^i_k, \omega^i_k\}_{i=1}^{N_P}\)

END IF

• Knowledge of the initial posterior is required – not possible, a proposal density is used
Particle filter
Sample Impoverishment Problem

- No degeneracy problem, but new problem arises
- Particle with too high weight are selected more and more often
- The other ones will slowly die out
- This leads to a loss of diversity or *sample impoverishment*
- For small process noise, all particles can collapse into a single point within a few iterations
- Another problem: resampling limits the possibility to parallelize the algorithm
RF-based Localization
Position tracking – Bayesian estimation methods

• There are various other kinds of particle filters in the literature
  − sampling importance resampling (SIR)
  − auxiliary sampling importance resampling (ASIR)
  − regularized particle filter (RPF)
  − Gaussian sum particle filter (GSPF)
  − kernel density estimation based
  − ...

• There are also Markov Chain Monte Carlo (MCMC) batch Bayesian methods

• They model the full posterior $x_k \sim p(x_0, \ldots, x_k | y_k, \ldots, y_0)$ – not online, very complex
Bayesian filters
Pros. and cons.

✔ close to optimal performance
✔ can deal with non-linearities, non-Gaussian noise – even multimodal distributions are not a problem
✔ PF can be implemented in $O(N_P)$, mostly parallelizable
✔ in contrast to HMM filters (state space discretized to $N_S$) states, PF focus on probable regions of the state space
✔ Bayesian filters can reliably fuse data originating from different sensors (e.g. RF signals, kinematic sensors, cameras, etc.)
✔ As long as the sources of measurement error (randomness) are independent, fusing multiple sensors will always improve results
✘ might be too complex for some real-time applications
✘ degeneracy and sample impoverishment problems may arise
✘ choosing the importance density might be hard
Bayesian Filters

Thesis: “If you want to solve a problem, particle filters are the best filters you can use, much better than e.g. Kalman filters.”

- Right or wrong?
- Very wrong!
- PF contain randomness – only converge to the true posterior when number of particles $N_p \to \infty$ (asymptotically optimal)
- If assumption for Kalman filters or grid-based filters hold, no particle filter can outperform them
- In case these assumptions hold approximately, other filters may perform reasonably well with much lower complexity than PF
- For location estimation and tracking, these assumptions do not hold, in general (multimodal p.d.f’s arise easily)
Practical aspects for localization

Motion model

- It is important to have an accurate description of the reality
- People’s motion is subject to physical constraints
  - position (accessible areas, transition between them)
  - speed (stopped, walking, running)
  - acceleration (changes in speed are limited)
- It is sensible to take into account these constraints when doing position tracking
- Motion models predict the next position based on the previous one
- Information about the building layout may be used as well
- Such information may improve drastically the location estimation and tracking
- There are many motion models in the literature: uniform linear motion, diffusion approach, angular p.d.f., multi-user motion models, etc. [AbrPauBar11, Roy11, KaiKhi11, QuiSta10]
Real-world results

Work in Progress...

• We test a grid-based filter using real-world results, the same as in [AbrPauBar11]
• The sensor nodes are equipped with two RF front-ends operating at 433MHz and 2.4GHz, respectively
• RSSI on both frequency was used
• no knowledge about the building geometry
• unknown starting point (initial posterior based on the first measurement)
• the same linear motion model as in [AbrPauBar11]
• the true position was measured using a laser range meter
Real-world results
Work in Progress...

Indoor location estimation and tracking

- office environment in Aveiro
- grid-based filter
- spatial grid resolution: 0.5 meters
- single user case (for better visualization)
- also because of limited amount of measurement data
- true user path – gray solid line with circle markers
- anchor nodes: $A_1, \ldots, A_4$

Figure 5: Location estimation and tracking
Real-world results

Work in Progress...

- Receive the very first measurement $z_1$
- initial (proposal) posterior distribution $p(x_1|z_{1:1})$ – continuous representation
- computed just using the first measurement $z_1$

Figure 6: Location estimation and tracking
Real-world results
Work in Progress...

- Prediction step – get the prior:

\[ p(x_2 | z_{1:1}) = \int_{D_x} p(x_2 | x_1) p(x_1 | z_{1:1}) dx_1 \]

- \( p(x_2 | x_1) \): based on the prediction equation
- \( p(x_1 | z_{1:1}) \): the old posterior

- The spreading of the posterior after the prediction step may be noticed

Figure 7: Location estimation and tracking
Real-world results

Work in Progress...

- receive new measurement $z_2$
- Compute its likelihood $p(z_2|x_2)$ based on the measurement equation (log-distance RSSI channel model)
- Multiply the likelihood with the prior $p(x_2|z_1:1)$ and normalize to get the new posterior

Figure 8: Location estimation and tracking
Real-world results
Work in Progress...

- New posterior
  \[ p(x_2 | z_{1:2}) = \frac{p(z_2 | x_2) p(x_2 | z_{1:1})}{p(z_2 | z_{1:1})} \]

- The denominator is just a normalization factor – it does not need to be calculated

- Shrinking of the new posterior may be noticed

- Calculate new position estimate \( \hat{x}_2 \), as the dominant mode (maximum \textit{a posteriori} estimate)

Figure 9: Location estimation and tracking
Real-world results
Work in Progress...

Figure 10: Location estimation and tracking

- The estimated (red line) vs. the true position (gray line)
- Progress is shown by the orange line
- Repeat this procedure at every step...
Real-world results
Work in Progress...

Figure 11: Location estimation and tracking

• Prediction: $p(x_3 | z_{1:2})$
Real-world results

Work in Progress...

\[ p(x_3 \mid r_{1:2}) \times p(r_3 \mid x_3) \]

- New measurement and its likelihood \( p(z_3 \mid z_3) \) to be multiplied with the prior

Figure 12: Location estimation and tracking
Real-world results

Work in Progress...

Figure 13: Location estimation and tracking

\[ p(x_3 | r_{1:3}) \]

- New posterior \( p(x_3 | z_{1:3}) \)
Real-world results
Work in Progress...

Figure 14: Location estimation and tracking

• New position estimate $\hat{x}_3$
• and so on...

$\begin{align*}
x_{\text{est}}(3) & \text{ vs. } x_{\text{true}}(3) \\
\end{align*}$
Real-world results
Work in Progress...

Figure 15: Location estimation and tracking

- Tracked location (red solid line with circle markers)
Real-world results

Work in Progress...

Figure 16: Location estimation and tracking

• The discretized position estimates

West–East axis [meters]

South–North axis [meters]

$x_{est}(25)$ vs. $x_{true}(25)$
Conclusions

• Bayesian filters represent powerful tools for location estimation and tracking

• Various approaches exist (KF, EKF, UKF, grid-based filter, particle filters...)

• They have different degrees complexity/accuracy

• There is no “best option”, the choice depends on the application at hand
References


References


References

