Budapesti Műszaki és Gazdaságtudományi Egyetem

Közlekedés- és Járműirányítási Tanszék

Automotive Environment Sensing

03 – Localization with particle filter

Olivér Törő

2019

Related concepts

Concepts related to vehicles moving in an environment:

- State estimation
- Localisation
- Mapping
- SLAM (simultaneous localization and mapping)
- Navigation, motion planning

General term describing this field: **Robot Mapping**

Related concepts

- **State estimation**: in this context the state to be estimated is the position (and orientation) of the vehicle and the positions of the landmarks or other features that are recognized in the environment
- **Localizaton**: determining the position of the vehicle on a given map
- **Mapping**: bulding a map of the environment with known vehicle positions
- **SLAM**: simultaneous localization and mapping, vehicle position and map is unknown
- **Navigation**: given a map the vehicle autonomously chooses a path and reaches the destination

Localization

Scenario:

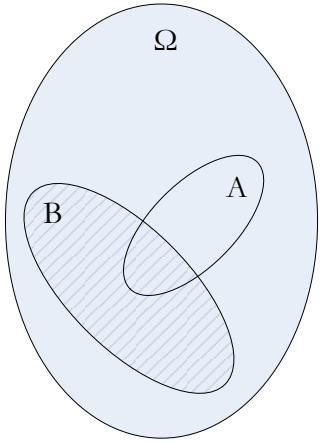
- The map is known
- Vehicle position is unknown
- We can perform measurements that give us information about the environment
- We have a model of the vehicle motion and the measurement process
- Find and track the vehicle on the map

Bayes formalism

- **Conditional probability** (definition)

$$P(A|B) := \frac{P(A \cap B)}{P(B)}$$

$$P(A \cap B) = P(A|B)P(B) = P(B|A)P(A)$$



- **Independent events**

$$P(A|B) = P(A) \qquad P(B|A) = P(B)$$

$$P(A \cap B) = P(A)P(B)$$

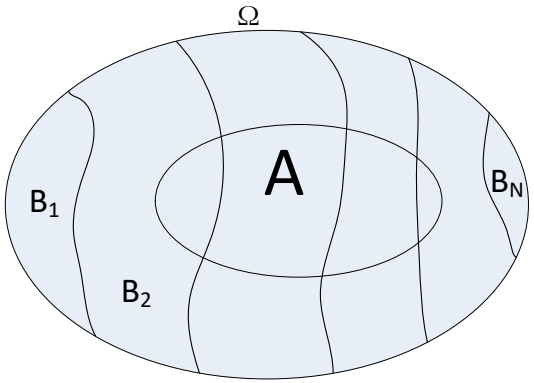
- **Collectively exhaustive events**

$$\bigcup_{i=1}^{N} B_i = \Omega \qquad B_i \cap B_j = \emptyset$$

Bayes formalism

- **Law of total probabilities:**

$$P(A) = \sum_{i=1}^{N} P(A \cap B_i) = \sum_{i=1}^{N} P(A \mid B_i) P(B_i)$$



- **Bayes theorem:**

$$P(B_k|A) = \frac{P(A|B_k)P(B_k)}{P(A)} = \frac{P(A|B_k)P(B_k)}{\sum_{i=1}^{N} P(A|B_i)P(B_i)}$$

Usual terminology:

Posterior: $P(B_k \mid A)$

Likelihood: $P(A \mid B_k)$

Prior: $P(B_k)$

Evidence, marginal likelihood: $P(A)$

Bayes formalism (notations)

- Position of the vehicle at time t is x_t
- Trajectory of the vehicle is the sequence of its positions in time: $\{x_0, x_1, x_2, \dots, x_t\}$ short notation is $x_{0:t}$ or x^t
- Environment sensors produce the measurement z_t
- We want to estimate
  - $p(x_{1:t} | z_{1:t})$ trajectory
  - $p(x_t | z_{1:t})$ actual position

Bayes estimation

$$p(x_t|z_{1:t}) = \frac{p(z_t|x_t)p(x_t|z_{1:t-1})}{p(z_t|z_{1:t-1})}$$

Can be computed in a recursive form:

- $p(z_t|x_t)$ likelihood, can be constructed from sensor model
- $p(z_t|z_{1:t-1})$ evidence or normalizing factor given by the integral:

$$p(z_t|z_{t-1}) = \int p(z_t|x_t)\, p(x_t|z_{t-1})\mathrm{d}x_t$$

- $p(x_t|z_{1:t-1})$ the prior is given by the time prediction integral:

$$p(x_t|z_{1:t-1}) = \int p(x_t|x_{t-1})\, p(x_{t-1}|z_{1:t-1})\mathrm{d}x_{t-1}$$

motion model previous posterior

Bayes estimation

- Computing the integrals is hard, generally no analytic solution exists thus we need to approximate
- In special cases exact solutions can be given: linear models and Gaussian noises leads to the Kalman filter
- In case of nonliear systems and arbitrary noise models we can use particle filters to evaluate the filtering equations

Monte Carlo methods

- Numerical computational techniques
- Construct a random process for the given problem
- Choose an adequate probability distribution
- Do lots of numerical experiments by sampling the random process

If a problem can be given a probabilistic interpretation, then it can be modelled using random numbers

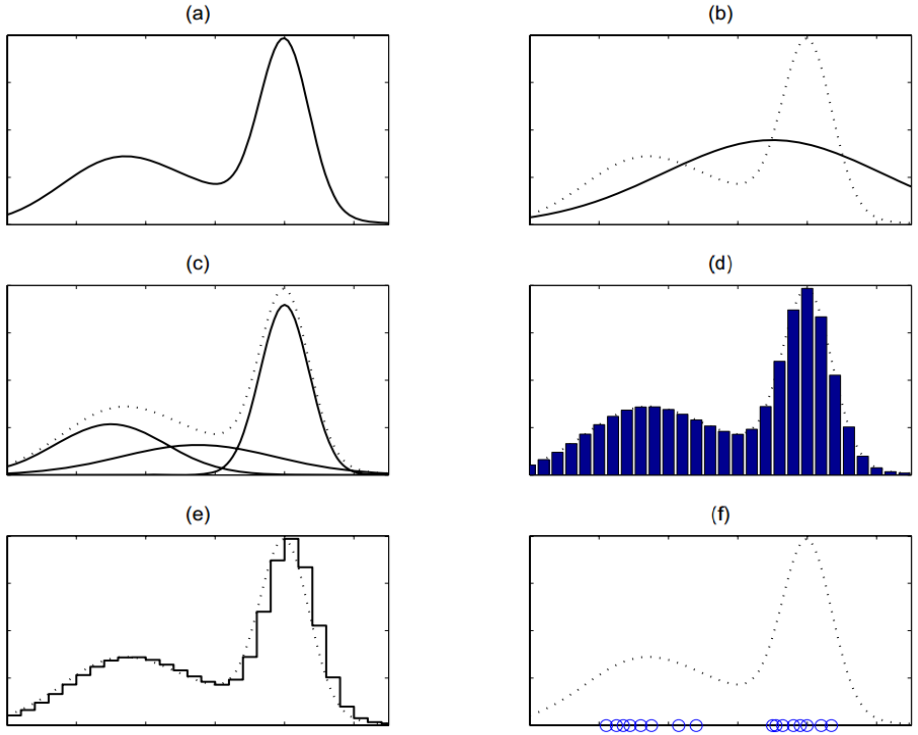
- Evaluate integrals
- Solve differential equations
- Simulate communication systems
- Study population dynamics

Particle filter

An arbitrary distribution can be approximated by particles, that are discrete samples taken from the distribution. Monte Carlo sampling

Approximating a distribution:

a) Original distribution
b) One Gaussian
c) Gaussian suml
d) Histogram
e) Step (Riemann) approx.
f) Monte Carlo sampling

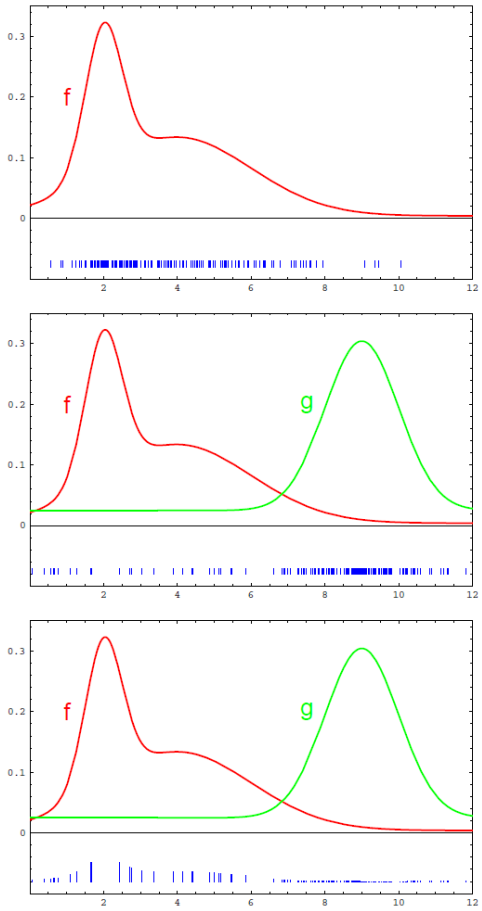


Particle filter

- Particle filter (PF): recursive algorithm for estimating in the Bayes formalism.
- Can handle multimodal distributions, no Gaussian noise constrain
- System model can be nonlinear
- Non-parametric estimator, that can approximate a distribution without a known analytic form
- The distribution is represented by a set of weighted samples (particles)
- The method is effective when the samples are choosen properly
- How to generate particles?
  - The function to be determined is the a posteriori which is unknown, can't sample from that
  - We choose an other function that can be sampled and does not differ much
  - The weights take the differences into account
  - We can always sample from a uniform distribution but it is not effective

Importance weights

- Function f should be approximated, but cannot be sampled
- Instead we sample g, which is called the importance density
- Every particle x gets weighted by the factor $\frac{f(x)}{g(x)}$
- In Bayes estimation f is the posterior and g is the prior
- In the simplest case (bootstrap particle filter) the weights are the likelihoods, which is given by the sensor model $p(z_t|x_t)$
- If g and f are similar then the weights differ little
- Try to choose an importance density that minimizes the variance of the weights



Bootstrap particle filter

One recursion step:

1. Input: $\left(x_{k-1}^{(i)}, z_k\right), i = 1 \ldots N$
2. Sample from the prior: $x_{k|k-1}^{(i)} \sim p(x_k|x_{k-1})$
3. Importance weights: $w_k^{(i)} = p\left(z_t \middle| x_{k|k-1}^{(i)}\right)$
4. Normalize weights: $\sum_{i=1}^{N} w_k^{(i)} = 1$
5. Resample N times with replacement from the prior pool $x_{k|k-1}^{(i)}$. Each particle is drawn with probability equal to its weight.
6. Output: $x_k^{(i)}$

Difficulties in particle filtering

- Curse of dimensionality
  - As the state space grows the number of particles needed to cover it grows exponentially
  - The likelihood function becomes a narrow spike and most of the particles will have negligible weight
- Particle degeneration
  - We want to approximate a continuous PDF but in the resampling step we draw particles from a discrete pool. We need to introduce some variance.
- Precise measurement
  - It also causes a narrow likelihood function. Can happen that all particle will get zero weight.

Dealing with difficulties

- In case of noise free sensors (lidar) add artificial noise to the measurements
- Resample only when the effective sample size is under a treshhold

$$N_{eff} = \frac{1}{\sum(w_i)^2}$$

- After resampling perform kernel density estimation
  - Epanechnikov kernel is optimal in a mean square error sense

References

- **Robot Mapping:** course at University of Freiburg (online videos) http://ais.informatik.uni-freiburg.de/teaching/ws13/mapping/

- **Probabilistic Robotics (Sebastian Thrun, 2006): book** http://www.probabilistic-robotics.org/