Lecture 25, Mar 11, 2026

Filter-Based Localization

- Suppose we have a motion model $\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{v}_k, \boldsymbol{w}_k)$ for state $\boldsymbol{x}_k$, input $\boldsymbol{v}_k$ and process noise $\boldsymbol{w}_k$; we also have a measurement model $\boldsymbol{y}_k = \boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{n}_k)$ for some measurement noise $\boldsymbol{n}_k$; we want to produce an estimate $\hat{\boldsymbol{x}}_k$ of the true state, given the measurements, inputs, and models
- We assume that the process is *Markovian* (Markov assumption): the conditional PDFs of future states depends on the present state only, and is independent of any past states
  - This essentially means that our state contains enough information so that predicting the future states does not rely on the past states
- Our goal is to compute $p(\boldsymbol{x}_k | \check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k}, \boldsymbol{y}_{0:k})$, known as the *belief function* for $\boldsymbol{x}_k$
  - Using Bayes' rule: $p(\boldsymbol{x}_k | \check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k}, \boldsymbol{y}_{0:k}) = \eta p(\boldsymbol{y}_k | \boldsymbol{x}_k) p(\boldsymbol{x}_k | \check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k}, \boldsymbol{y}_{0:k-1})$ for some scaling factor η
    - Note that we can do this since the measurements $\boldsymbol{y}$ are conditionally independent given $\boldsymbol{x}$
  - The second term is $$\begin{aligned} p(\boldsymbol{x}_k | \check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k}, \boldsymbol{y}_{0:k-1}) &= \int p(\boldsymbol{x}_k, \boldsymbol{x}_{k-1} | \check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k}, \boldsymbol{y}_{0:k-1}) \, \mathrm{d}\boldsymbol{x}_{k-1} \\ &= \int p(\boldsymbol{x}_k | \boldsymbol{x}_{k-1}, \check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k}, \boldsymbol{y}_{0:k-1}) p(\boldsymbol{x}_{k-1} | \check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k}, \boldsymbol{y}_{0:k-1}) \, \mathrm{d}\boldsymbol{x}_{k-1} \end{aligned}$$
    - We've introduced the hidden state $\boldsymbol{x}_{k-1}$ through essentially reverse marginalization
  - Since the state has no dependence on future inputs, $p(\boldsymbol{x}_{k-1} | \check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k}, \boldsymbol{y}_{0:k-1}) = p(\boldsymbol{x}_{k-1} | \check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k-1}, \boldsymbol{y}_{0:k-1})$
  - Due to the Markov assumption, $p(\boldsymbol{x}_k | \boldsymbol{x}_{k-1}, \check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k}, \boldsymbol{y}_{0:k-1}) = p(\boldsymbol{x}_k | \boldsymbol{x}_{k-1}, \boldsymbol{v}_k)$
  - Finally, $p(\boldsymbol{x}_k | \check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k}, \boldsymbol{y}_{0:k}) = \eta p(\boldsymbol{y}_k | \boldsymbol{x}_k) \int p(\boldsymbol{x}_k | \boldsymbol{x}_{k-1}, \boldsymbol{v}_k) p(\boldsymbol{x}_{k-1} | \check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k-1}, \boldsymbol{y}_{0:k-1}) \, \mathrm{d}\boldsymbol{x}_{k-1}$
    - Note that this takes on a familiar predictor-corrector form: the prior $p(\boldsymbol{x}_{k-1} | \check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k-1}, \boldsymbol{y}_{0:k-1})$ is first used to generate a prediction using the motion model through $p(\boldsymbol{x}_k | \boldsymbol{x}_{k-1}, \boldsymbol{v}_k)$, then corrected using $p(\boldsymbol{y}_k | \boldsymbol{x}_k)$, from the measurement model and current observation
- However, the Bayes filter is unimplementable, since arbitrary PDFs are infinite dimensional and the integral is also impossible to compute without additional assumptions
  - To make this tractable, we must make approximations:
    - Analytical models – approximate the PDFs as some known type with a finite number of parameters, e.g. Gaussians
      - With the Gaussian approximation, this leads to the plain Kalman Filter, Extended Kalman Filter, and Unscented Kalman Filter
    - Histograms – discretize the state space and record a value for each cell
    - Particles – represent the distribution with a large number of hypotheses
      - This leads to particle filtering
- We will use $\check{\boldsymbol{x}}$ to denote the state prediction before the measurement is incorporated and $\hat{\boldsymbol{x}}$ to denote the prediction after the measurement is fused

Particle Filtering

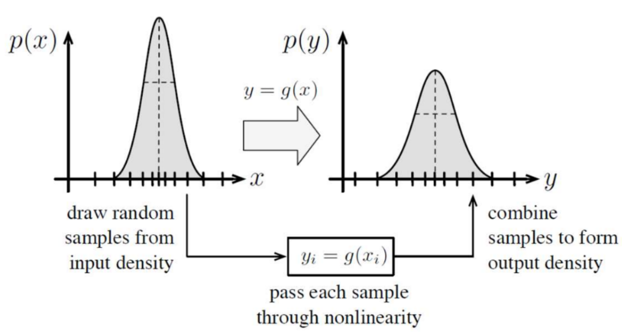

Figure 1: Monte Carlo sampling for particle filtering.

- *Particle filtering* represents the distribution with particles; we maintain many hypotheses for the robot's localization, and propagate them through our models

- Given enough computational resources this is able to handle any arbitrary PDF and nonlinearities in the models, since we need no assumptions
- We don't need analytical expressions for the motion or observation model as long as we can compute them
- Particle filter procedure:
  1. Prediction step:
     - Draw M samples $\begin{bmatrix} \check{\boldsymbol{x}}_{k-1,m} \\ \boldsymbol{w}_{k,m} \end{bmatrix}, m = 1, \ldots, M$ from the joint density with both the prior and motion noise, $p(\boldsymbol{x}_{k-1}|\check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k-1}, \boldsymbol{y}_{0:k-1})p(\boldsymbol{w}_k)$
     - Generate a prediction of the posterior by propagating each particle through the model: $\check{\boldsymbol{x}}_{k,m} = \boldsymbol{f}(\hat{\boldsymbol{x}}_{k-1,m}, \boldsymbol{v}_k, \boldsymbol{w}_{k,m})$
  2. Correction step:
     - Assign each particle a weight $w_{k,m} = \dfrac{p(\check{\boldsymbol{x}}_{k,m}|\check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k}, \boldsymbol{y}_{0:k})}{p(\check{\boldsymbol{x}}_{k,m}|\check{\boldsymbol{x}}_0, \boldsymbol{v}_{1:k}, \boldsymbol{y}_{0:k-1})} = \eta p(\boldsymbol{y}_k|\check{\boldsymbol{x}}_{k,m})$ where η is some normalizing scalar
       * In practice we simulate an expected measurement $\check{\boldsymbol{y}}_{k,m} = \boldsymbol{g}(\check{\boldsymbol{x}}_{k,m}, \boldsymbol{0})$, and then assume $p(\boldsymbol{y}_k|\check{\boldsymbol{x}}_{k,m}) = p(\boldsymbol{y}_k|\check{\boldsymbol{y}}_{k,m})$ where $p(\boldsymbol{y}_k|\check{\boldsymbol{y}}_{k,m})$ is some known density we derive from the measurement model
     - Resample the particles probabilistically based on the weight to get $\hat{\boldsymbol{x}}_{k,m}$, so particles with higher weight has a higher probability to get sampled
       * Several different ways exist but *Madow systematic sampling* is one effective and simple way
       * Create M bins based on cumulative weights, with the size of bin m being proportional to $w_{k,m}$; then starting at some random point, march along in steps of $1/M$ and sample from each bin that is visited
         - This has the advantage that if a particle has weight more than $1/M$, it is guaranteed to be sampled, which combats sample depletion; particles with lower weights are only sometimes sampled due to the random starting point
- Particle diversity (variance) is important in particle filters, since if none of the particles accurately models the true state, then we will never converge
  - As we resample, some particles will get missed due to the probabilistic sampling, and over time particles might deplete in diversity
  - Often we might want to simulate more noise than there actually is in the system, to maintain sufficient particle diversity; paradoxically having a noise variance that is too low will often hurt the filter performance
  - During the prediction step we may want to add particles uniformly drawn from the entire sample space to recover from sample impoverishment
  - We may want to resample less often to avoid this as well
- For low dimensional problems a few hundred or thousand particles might be enough, but this increases exponentially with dimensionality
  - Particle count can be picked online using a heuristic based on the sum of weights; if the sum of weights is low (i.e. all of the hypotheses are uncertain) then we might want to add more particles