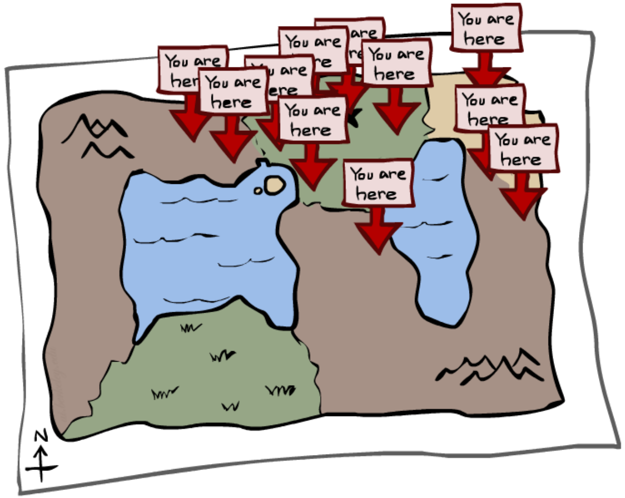
COMS W4733: Computational Aspects of Robotics

Lecture 21: Kalman and Particle Filters



Instructor: Tony Dear

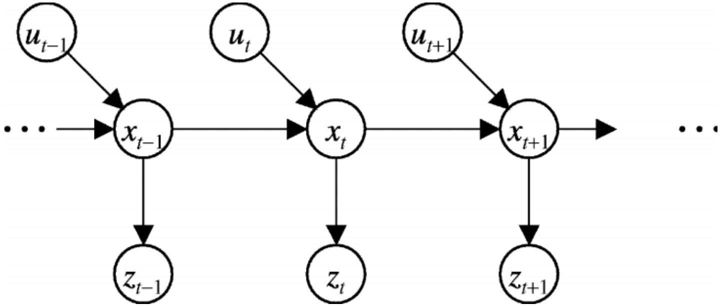
State Estimation

- Belief distribution $B(\boldsymbol{x}_t) = p(\boldsymbol{x}_t \mid \boldsymbol{u}_{1:t}, \boldsymbol{z}_{1:t})$

Robot state at time t Actions from 1 to t Observations from 1 to t

- Transition model $p(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{u}_t)$

- Observation model $p(\boldsymbol{z}_t \mid \boldsymbol{x}_t)$



Bayes Filter Algorithm

Algorithm **Bayes_filter**($B(\boldsymbol{x}), \boldsymbol{d}$):

1. $\eta = 0$
2. **if** $\boldsymbol{d}$ is an *action* data item $\boldsymbol{u}$ **then**
3. **for all** x **do**
4. $\overline{B}(\boldsymbol{x}) = \int p(\boldsymbol{x}|\boldsymbol{x}', \boldsymbol{u}) B(\boldsymbol{x}') d\boldsymbol{x}'$
5. **if** $\boldsymbol{d}$ is a *perceptual* data item $\boldsymbol{z}$ **then**
6. **for all** x **do**
7. $\overline{B}(\boldsymbol{x}) = p(\boldsymbol{z}|\boldsymbol{x}) B(\boldsymbol{x})$
8. $\eta = \eta + \overline{B}(\boldsymbol{x})$
9. **for all** x **do**
10. $\overline{B}(\boldsymbol{x}) = \eta^{-1} \overline{B}(\boldsymbol{x})$
11. **return** $\overline{B}(\boldsymbol{x})$

Prediction:

$$B'(\boldsymbol{x}_t) = \int \underbrace{p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{u}_t)}_{\text{Transition model}} B(\boldsymbol{x}_{t-1})\, d\boldsymbol{x}_{t-1}$$

Observation:

$$B(\boldsymbol{x}_t) = \eta^{-1}\, \underbrace{p(\boldsymbol{z}_t|\boldsymbol{x}_t)}_{\text{Observation model}} B'(\boldsymbol{x}_t)$$

Bayes Filter Considerations

- Bayes filter is a recursive algorithm that computes robot's posterior belief given prior belief and either an action or observation

- Problems: Prediction step requires integration of transition model
- Normalization after observation step also requires an integration over entire belief distribution

- Typically very difficult to do analytically, very expensive to do numerically
- What if our distributions are all Gaussian?

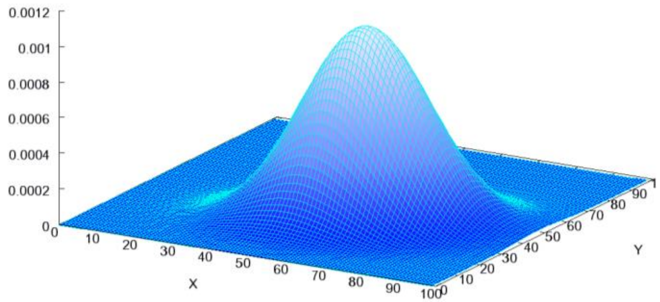
Gaussian Distributions

- A multivariate Gaussian distribution has two parameters: mean vector $\boldsymbol{\mu}$, covariance matrix $\boldsymbol{\Sigma}$

$$B(\boldsymbol{x}_t) = p(\boldsymbol{x}_t \mid \boldsymbol{u}_{1:t}, \boldsymbol{z}_{1:t}) = \frac{1}{\sqrt{(2\pi)^{|x_t|}|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\boldsymbol{x}_t - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}_t - \boldsymbol{\mu})\right)$$

- Suppose our belief distributions stay Gaussian while being propagated through the Bayes filter

- A **Kalman filter** computes analytical updates for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ and avoids expensive integrations!



Gaussian Affine Transformations

- We have Gaussian random variables $\boldsymbol{X} \sim N(\boldsymbol{\mu_X}, \boldsymbol{\Sigma_X})$ and $\boldsymbol{Y} \sim N(\boldsymbol{\mu_Y}, \boldsymbol{\Sigma_Y})$

- If $\boldsymbol{A}$ and $\boldsymbol{B}$ are constant matrices and $\boldsymbol{C}$ is a constant vector, $\boldsymbol{AX} + \boldsymbol{BY} + \boldsymbol{C}$ remains a Gaussian random variable

- Mean (same transformations as on RVs): $\boldsymbol{A\mu_X} + \boldsymbol{B\mu_Y} + \boldsymbol{C}$
- Covariance (no covariance from $\boldsymbol{C}$, covariances of $\boldsymbol{X}$ and $\boldsymbol{Y}$ are rotated and summed): $\boldsymbol{A\Sigma_X A^T} + \boldsymbol{B\Sigma_Y B^T}$

$$\boldsymbol{AX} + \boldsymbol{BY} + \boldsymbol{C} \sim N(\boldsymbol{A\mu_X} + \boldsymbol{B\mu_Y} + \boldsymbol{C}, \boldsymbol{A\Sigma_X A^T} + \boldsymbol{B\Sigma_Y B^T})$$

Product of Gaussians

- We have Gaussian random variables $\boldsymbol{X} \sim N(\boldsymbol{\mu_X}, \boldsymbol{\Sigma_X})$ and $\boldsymbol{Y} \sim N(\boldsymbol{\mu_Y}, \boldsymbol{\Sigma_Y})$
- Their (normalized) product is also a Gaussian
- **Gain matrix**:

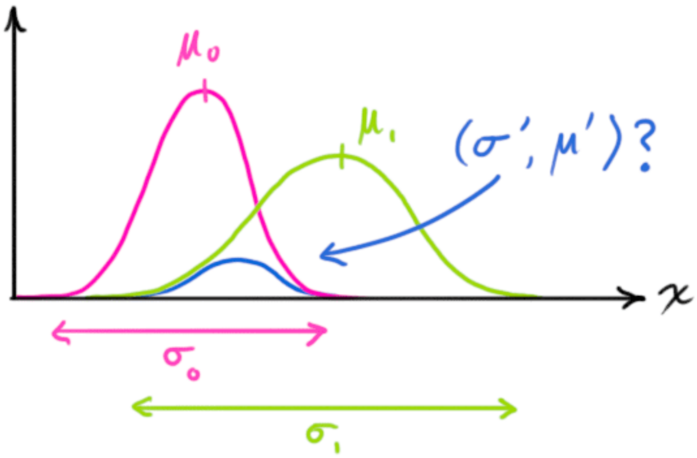
$$\boldsymbol{K} = \boldsymbol{\Sigma_X}(\boldsymbol{\Sigma_X} + \boldsymbol{\Sigma_Y})^{-1}$$

- **Mean** (add difference of means scaled by the gain matrix):

$$\boldsymbol{\mu_X} + \boldsymbol{K}(\boldsymbol{\mu_Y} - \boldsymbol{\mu_X})$$

- **Covariance** ("average" two covariances):

$$\boldsymbol{\Sigma_X} - \boldsymbol{K}\boldsymbol{\Sigma_X}$$



Model Assumptions

- In order for these Gaussian assumptions to hold, we'll need a few more requirements on our transition and observation models
- We assume both to be **linear** with **additive, Gaussian noise**
  - If models are nonlinear (most are!) we can either linearize them first, or use an **extended Kalman filter** to deal with them (later)

- Transition model

$$\boldsymbol{x}_k = \boldsymbol{F}_k \boldsymbol{x}_{k-1} + \boldsymbol{B}_k \boldsymbol{u}_k + \boldsymbol{w}_k \leftarrow \sim N(0, \boldsymbol{Q}_k)$$

State transition matrix

Input control matrix

- Observation model

$$\boldsymbol{z}_k = \boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{v}_k \leftarrow \sim N(0, \boldsymbol{R}_k)$$

Observation matrix

Transition Update

$$\boldsymbol{x}_k = \boldsymbol{F}_k \boldsymbol{x}_{k-1} + \boldsymbol{B}_k \boldsymbol{u}_k + \boldsymbol{w}_k$$

- Current belief state is Gaussian: $Bel(\boldsymbol{x}_{k-1}) \sim N(\hat{\boldsymbol{x}}_{k-1|k-1}, \boldsymbol{P}_{k-1|k-1})$
- How do mean and covariance update?

Current mean vector

Current covariance matrix

- Mean: Same transformation as transition model, no contribution from $\boldsymbol{w}_k$

$$\hat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{F}_k \hat{\boldsymbol{x}}_{k-1|k-1} + \boldsymbol{B}_k \boldsymbol{u}_k$$

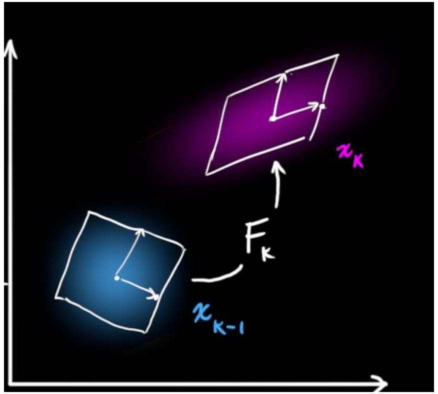
- Covariance: No contribution from $\boldsymbol{B}_k \boldsymbol{u}_k$, add $\mathrm{cov}(\boldsymbol{w}_k) = \boldsymbol{Q}_k$

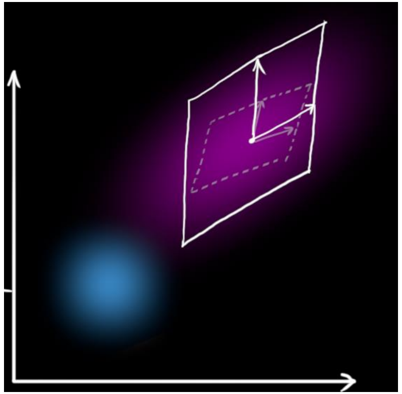
$$\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_k \boldsymbol{P}_{k-1|k-1} \boldsymbol{F}_k^T + \boldsymbol{Q}_k$$

Transition Update

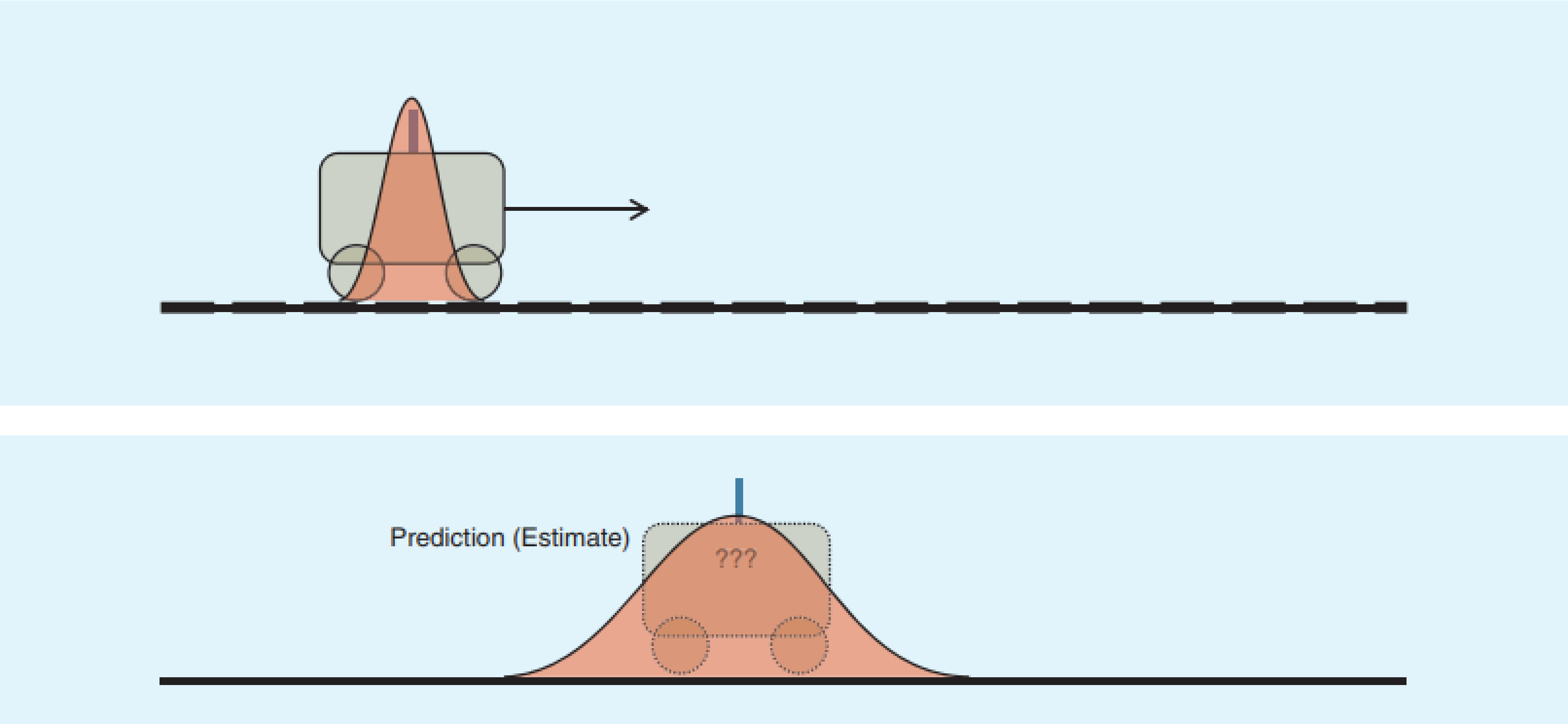
- Mean $\hat{\boldsymbol{x}}_{k|k-1}$ is shifted to where we think we are most likely to be (without noise, since zero mean) according to our transition model
- Covariance $\boldsymbol{P}_{k|k-1}$ is *rotated* in state space according to $\boldsymbol{F}_k$ and then *added to* (uncertainty increases) by $\boldsymbol{Q}_k$

$$\hat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{F}_k \hat{\boldsymbol{x}}_{k-1|k-1} + \boldsymbol{B}_k \boldsymbol{u}_k$$
$$\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_k \boldsymbol{P}_{k-1|k-1} \boldsymbol{F}_k^T + \boldsymbol{Q}_k$$





Example: Transition Update



Observation Update

- At our new belief state $\boldsymbol{x}_{k|k-1}$, we *expect* an observation that looks like $N(\boldsymbol{H}_k \boldsymbol{x}_{k|k-1}, \boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^T)$

- Suppose we actually observe $\boldsymbol{z}_k$ (with covariance $\boldsymbol{R}_k$)

- Multiply the two distributions together to get an updated posterior!

$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^T \left(\boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^T + \boldsymbol{R}_k \right)^{-1}$$

Kalman gain

$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k \underline{(\boldsymbol{z}_k - \boldsymbol{H}_k \hat{\boldsymbol{x}}_{k|k-1})}$$

Innovation

$$\boldsymbol{P}_{k|k} = \boldsymbol{P}_{k|k-1} - \boldsymbol{K}_k \boldsymbol{H}_k \boldsymbol{P}_{k|k-1}$$

Kalman Gain

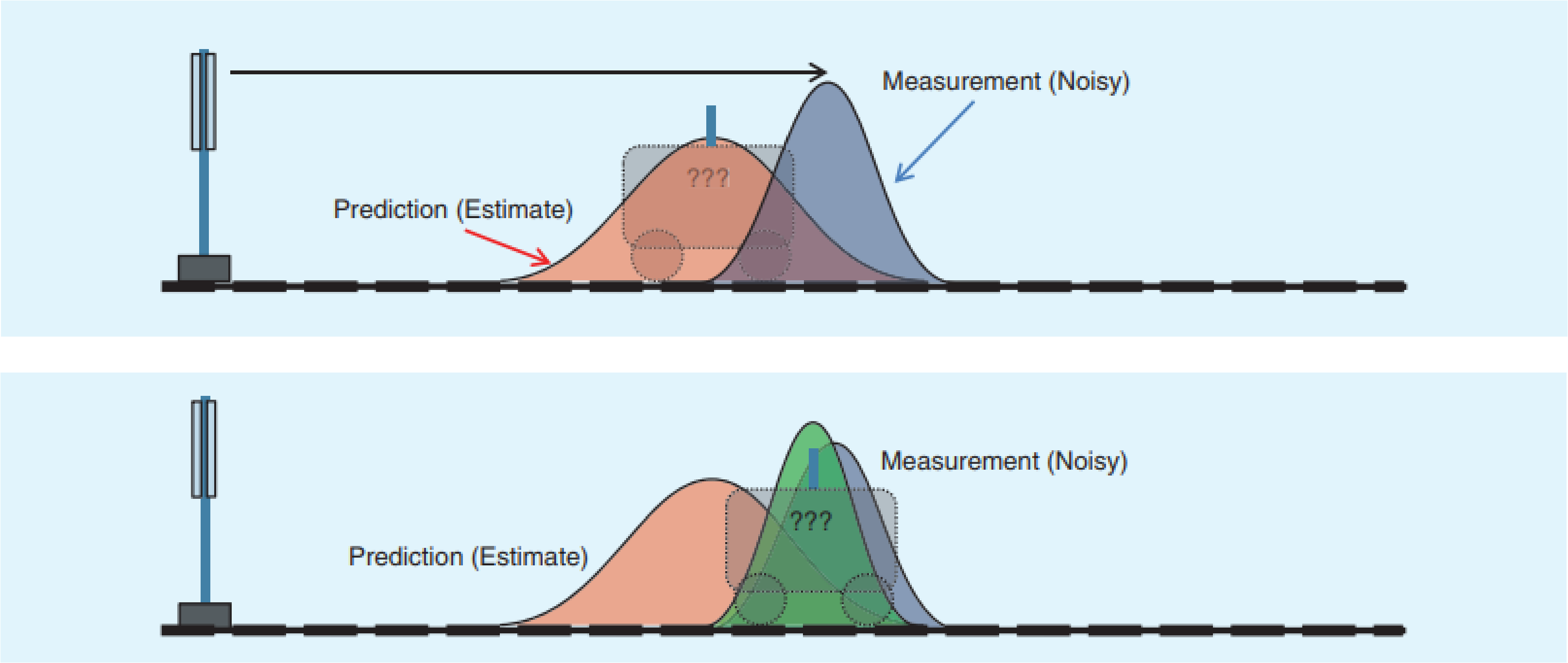
$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^T\left(\boldsymbol{H}_k\boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^T + \boldsymbol{R}_k\right)^{-1}$$

- Kalman gain tells us how much we want to update both mean and covariance
- $\boldsymbol{S}_k = \boldsymbol{H}_k\boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^T + \boldsymbol{R}_k$ is the covariance of the innovation $\boldsymbol{z}_k - \boldsymbol{H}_k\widehat{\boldsymbol{x}}_{k|k-1}$

- $\boldsymbol{R}_k \to 0, \boldsymbol{K}_k \to \boldsymbol{H}_k^{-1}$: Less uncertainty in measurement, trust observation more
- $\boldsymbol{P}_k \to 0, \boldsymbol{K}_k \to 0$: Less uncertainty in prediction, rely on observation less
- Conversely, as $\boldsymbol{P}_k$ goes up, we expect predictions to change more

- $\boldsymbol{K}_k$ varies inversely with $\boldsymbol{S}_k$, overall variability in measurement

Example: Observation Update



Kalman Filter

- Start with current belief distribution:

$$Bel(\boldsymbol{x}_{k-1}) \sim N(\widehat{\boldsymbol{x}}_{k-1|k-1}, \boldsymbol{P}_{k-1|k-1})$$

- Predict according to transition model:

$$\widehat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{F}_k \widehat{\boldsymbol{x}}_{k-1|k-1} + \boldsymbol{B}_k \boldsymbol{u}_k$$

$$\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_k \boldsymbol{P}_{k-1|k-1} \boldsymbol{F}_k^T + \boldsymbol{Q}_k$$

- Update according to observation model and measurement:

$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^T \left(\boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^T + \boldsymbol{R}_k \right)^{-1}$$

$$\widehat{\boldsymbol{x}}_{k|k} = \widehat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k (\boldsymbol{z}_k - \boldsymbol{H}_k \widehat{\boldsymbol{x}}_{k|k-1})$$

$$\boldsymbol{P}_{k|k} = \boldsymbol{P}_{k|k-1} - \boldsymbol{K}_k \boldsymbol{H}_k \boldsymbol{P}_{k|k-1}$$

Extended Kalman Filter

- What if our transition and/or observation models are nonlinear?

$$\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k) + \boldsymbol{w}_k$$

$$\boldsymbol{z}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{v}_k$$

- Then we just need to find Jacobians for $\boldsymbol{f}$ and $\boldsymbol{h}$ and equations remain mostly unchanged!

$$\boldsymbol{F}_k = \nabla f$$

$$\boldsymbol{H}_k = \nabla h$$

$$\hat{\boldsymbol{x}}_{k|k-1} = \boxed{f(\hat{\boldsymbol{x}}_{k-1|k-1}, \boldsymbol{u}_k)}$$

$$\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_k \boldsymbol{P}_{k-1|k-1} \boldsymbol{F}_k^T + \boldsymbol{Q}_k$$

$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^T \left(\boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^T + \boldsymbol{R}_k\right)^{-1}$$

$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k \left(\boldsymbol{z}_k - \boxed{h(\hat{\boldsymbol{x}}_{k|k-1})}\right)$$

$$\boldsymbol{P}_{k|k} = \boldsymbol{P}_{k|k-1} - \boldsymbol{K}_k \boldsymbol{H}_k \boldsymbol{P}_{k|k-1}$$

Kalman Filter Considerations

- KF is fast, analytic, optimal, recursive, and used in many applications
- E.g., vehicle navigation, computer vision, signal processing, econometrics, etc.
- Lots of real processes are linear and Gaussian (or close to Gaussian)!

- However, many robotics problems are nonlinear and possibly non-Gaussian
- EKF can handle nonlinearities but still require linear approximation (Jacobians)

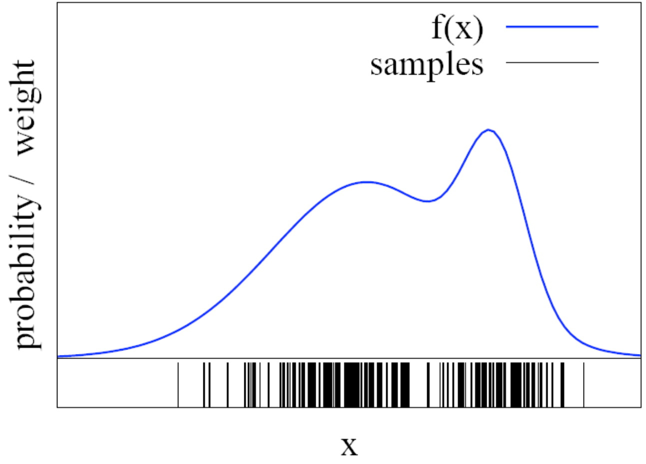
- If we can afford more computational power, we can use a sampling-based (Monte Carlo) approach to fit these belief distributions

Particle Filters

- Idea: *Approximate* belief distribution with a bunch of particles (samples)
- *Move* particles around according to our prediction (transition model)
- *Weight* particles according to our observations (observation model)
- *Resample* particles to obtain a new normalized distribution

$$B'(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_t | \boldsymbol{x}_{t-1}, \boldsymbol{u}_t) B(\boldsymbol{x}_{t-1})\, d\boldsymbol{x}_{t-1}$$

$$B(\boldsymbol{x}_t) = \eta^{-1}\, p(\boldsymbol{z}_t | \boldsymbol{x}_t) B'(\boldsymbol{x}_t)$$



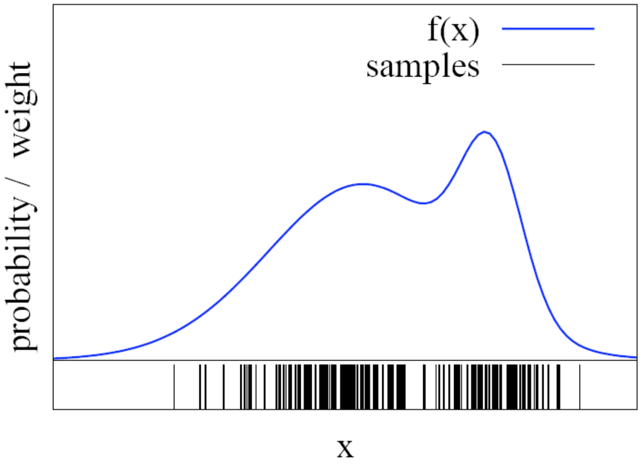
Particle Filter Algorithm

Algorithm **Particle_filter**($X_{t-1}, \boldsymbol{u}_t, \boldsymbol{z}_t$):

1. $\overline{X}_t = \emptyset, X_t = \emptyset$
2. **for** each particle x_{t-1}^j in X_{t-1} **do**
3. sample x_t^j from $p(\boldsymbol{x}_t | \boldsymbol{x}_{t-1}^j, \boldsymbol{u}_t)$
4. compute weight $w_t^j = p(\boldsymbol{z}_t | \boldsymbol{x}_t^j)$
5. insert (x_t^j, w_t^j) into $\overline{X}_t$
6. **for** all j **do**
7. sample $i \in \{1,2,\dots,J\}$ with prob $\frac{w_t^j}{\sum w_t^j}$
8. insert x_t^i from $\overline{X}_t$ into X_t
9. **return** X_t

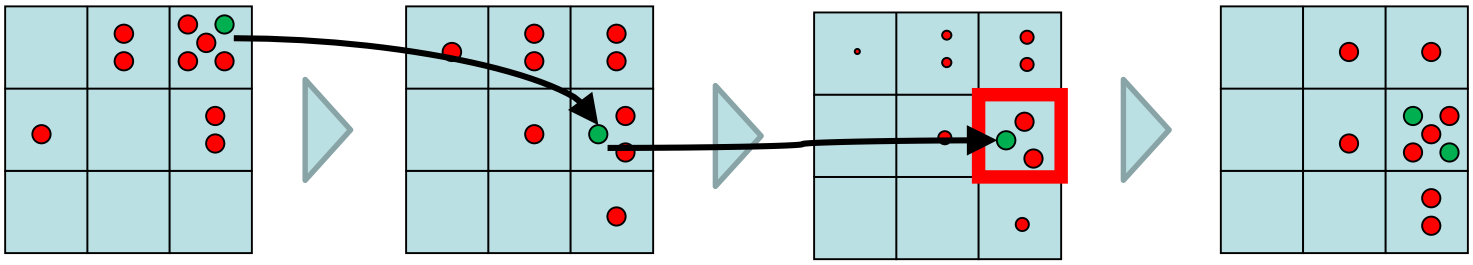
$$B'(\boldsymbol{x}_t) = \int p(\boldsymbol{x}_t | \boldsymbol{x}_{t-1}, \boldsymbol{u}_t) B(\boldsymbol{x}_{t-1})\, d\boldsymbol{x}_{t-1}$$

$$B(\boldsymbol{x}_t) = \eta^{-1} p(\boldsymbol{z}_t | \boldsymbol{x}_t) B'(\boldsymbol{x}_t)$$



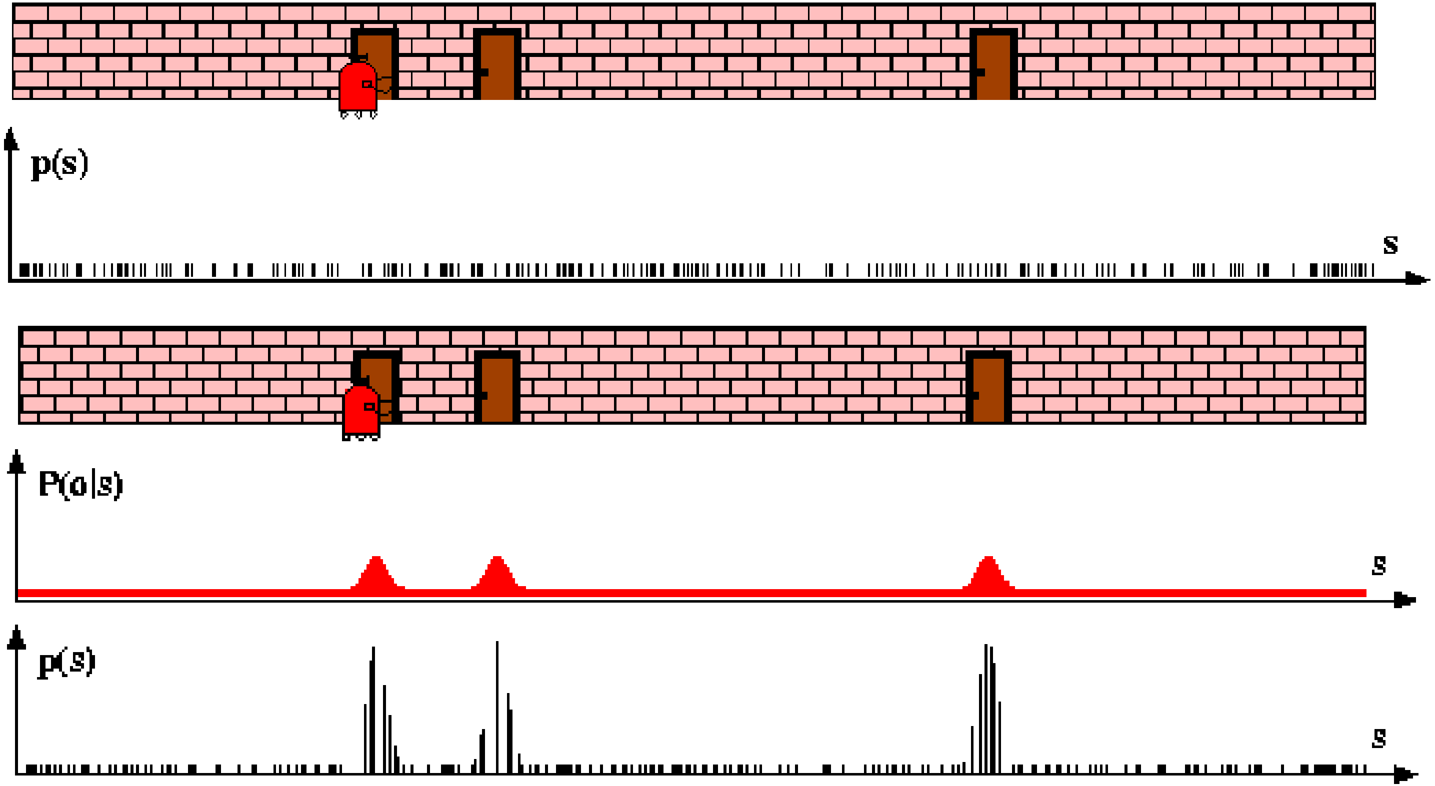
Particle Filter Example

sample $p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}^{j},\boldsymbol{u}_t)$ weight $w_t^j = p(\boldsymbol{z}_t|\boldsymbol{x}_t^j)$ Resample (renormalize):

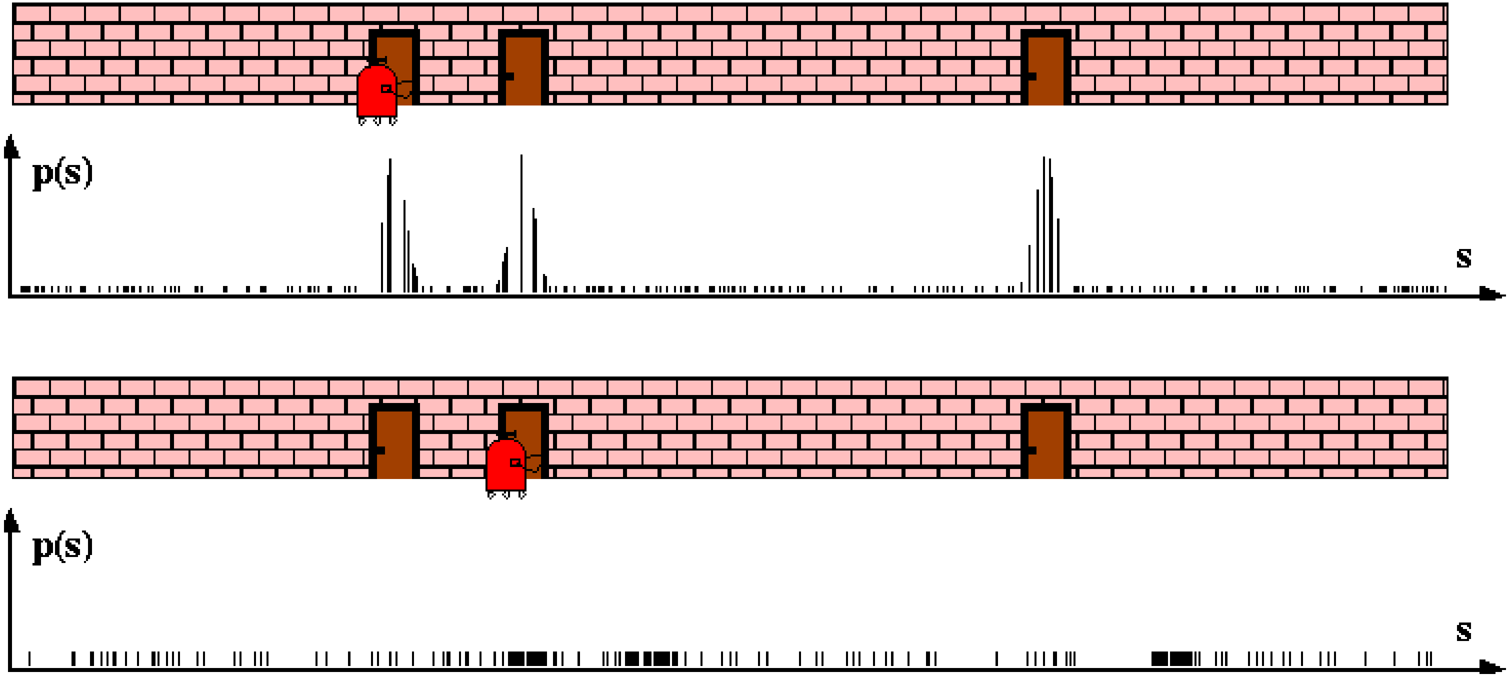


Particles:	Particles:	Particles:	(New) Particles:
(3,3)	(3,2)	(3,2) w=.9	(3,2)
(2,3)	(2,3)	(2,3) w=.2	(2,2)
(3,3)	(3,2)	(3,2) w=.9	(3,2)
(3,2)	(3,1)	(3,1) w=.4	(2,3)
(3,3)	(3,3)	(3,3) w=.4	(3,3)
(3,2)	(3,2)	(3,2) w=.9	(3,2)
(1,2)	(1,3)	(1,3) w=.1	(1,3)
(3,3)	(2,3)	(2,3) w=.2	(2,3)
(3,3)	(3,2)	(3,2) w=.9	(3,2)
(2,3)	(2,2)	(2,2) w=.4	(3,2)

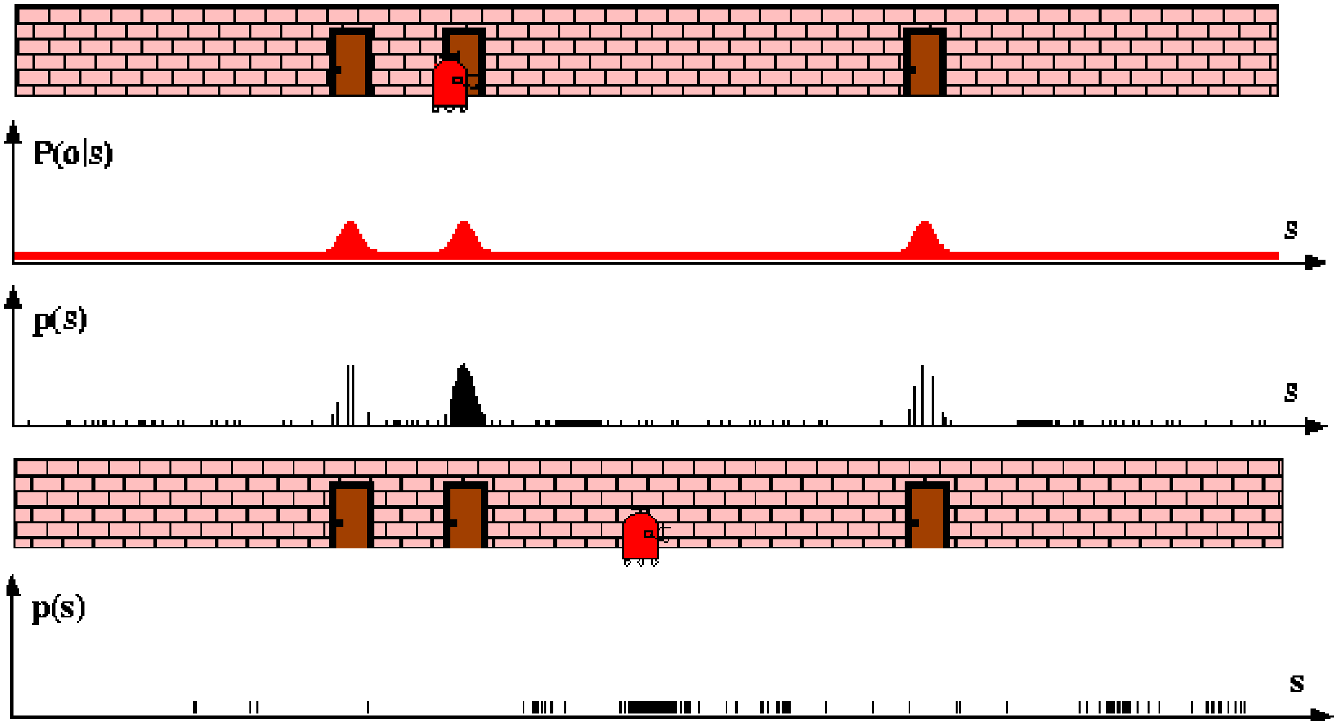
Example: Particle Filter



Example: Particle Filter

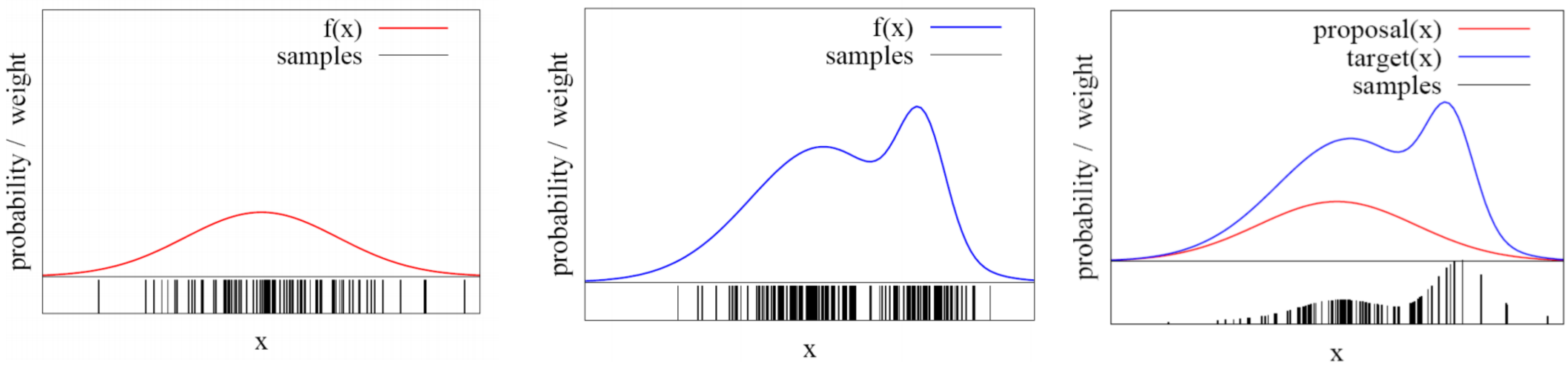


Example: Particle Filter



Importance Sampling

- Particle filter sampling process is a form of **importance sampling**
- Difficult to sample the posterior directly, so we sample from something we know (the prior) and then assign weights according to observations



Particle Filter Considerations

- Easy to implement, performance scales with number of particles
- Variations in resampling process—we don’t need to resample every step, especially when we don’t have any observations
- Resampling too often can lead to particle drift and *loss of diversity*

- If sensor noise is low, then measurement distribution will be narrow and highly peaked
- Problem: This will lead to low weights for many particles and zero them out
- *Particle deprivation* can happen if we are unlucky, when all particles in a given area are wiped out solely due to randomness

- For both these issues, consider introducing more noise or particles in the process

Summary

- Both Kalman and particle filters implement the Bayes filtering algorithm without having to explicitly compute and integrate over exact distributions

- Kalman filter: Everything is Gaussian; recursive, closed-form updates to distribution parameters (means, covariances)
- Key quantity: Kalman gain indicates strength of updates

- Particle filter: Any distribution goes, can approximate it with discrete samples
- Inference updates implemented via importance sampling and resampling

- https://www.bzarg.com/p/how-a-kalman-filter-works-in-pictures/