



Autonomous Robotics

Spring 2026

Abhishek Gupta, Siddhartha Srinivasa

TAs: Helen Wang, Sidharth Talia, Rohan Baijal, Christopher Tan



Recap

Problem 1: Two Room Challenge

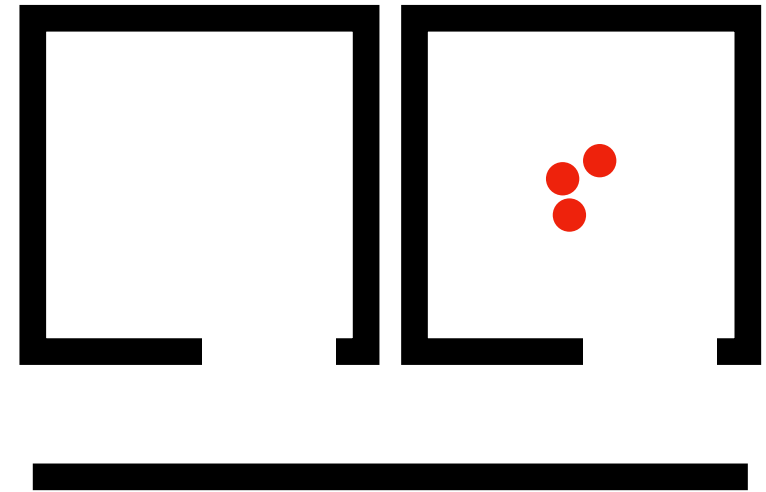
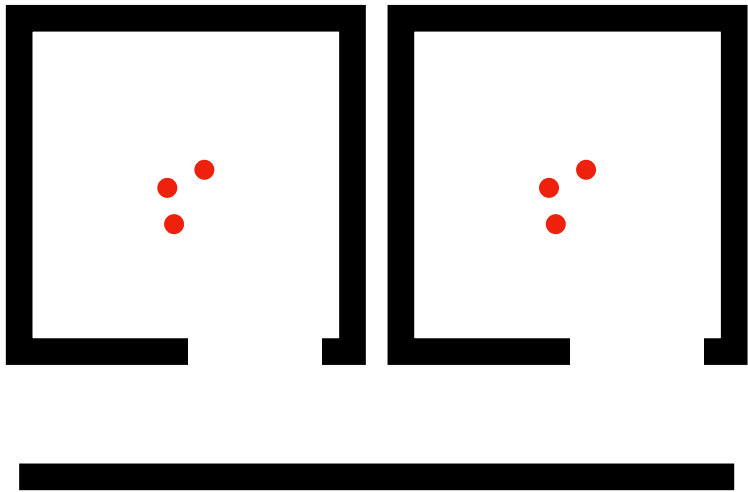
Particles begin equally distributed, no motion or observation



All particles migrate to one room!

Reason: Resampling Increases Variance

50% prob. of resampling particle from Room 1 vs Room 2
31% prob. of preserving 50-50 particle split



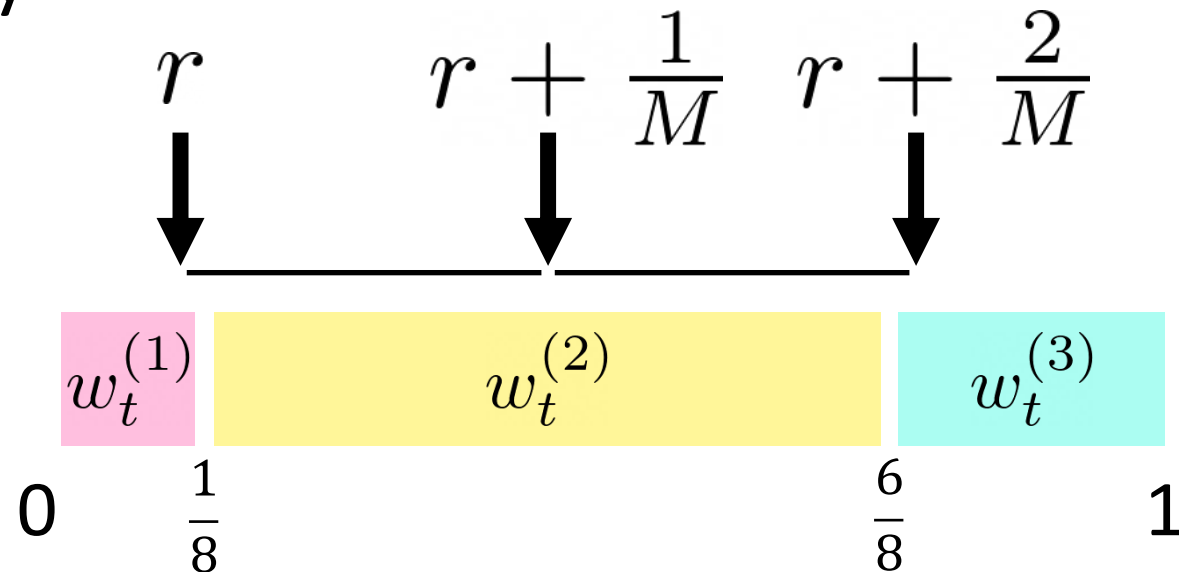
All particles migrate to one room!

Idea 1: Judicious Resampling

- Key idea: resample less often! (e.g., if the robot is stopped, don't resample). Too often may lose particle diversity, infrequently may waste particles
- Common approach: don't resample if weights have low variance
- Can be implemented in several ways: don't resample when...
 - ...all weights are equal
 - ...weights have high entropy
 - ...ratio of max to min weights is low

Idea 2: Low-Variance Resampling

- Sample one random number $r \sim [0, \frac{1}{M}]$
- Covers space of samples more systematically (and more efficiently)
- If all samples have same importance weight, won't lose particle diversity



Other Practical Concerns

- How many particles is enough?
 - Typically need more particles at the beginning (to cover possible states)
 - [KLD Sampling \(Fox, 2001\)](#) adaptively increases number of particles when state uncertainty is high, reduces when state uncertainty is low
- Particle filtering with overconfident sensor models
 - Squash sensor model prob. with power of $1/m$
 - Sample from better proposal distribution than motion model
 - [Manifold Particle Filter \(Koval et al., 2017\)](#) for contact sensors
- Particle starvation: no particles near current state

MuSHR Localization Project

- Implement kinematic car motion model
- Implement different factors of single-beam sensor model
- Combine motion and sensor model with the Particle Filter algorithm

Lecture Outline

Recap



Kalman Filtering

Can we get closed form updates for Bayesian Filtering?

Need to choose form of probability distributions

- Dynamics (Prediction)

$$\overline{Bel}(x_t) = \int p(x_t | u_t, x_{t-1}) Bel(x_{t-1}) dx_{t-1}$$

- Measurement (Correction)

$$Bel(x_t) = \eta P(z_t | x_t) \overline{Bel}(x_t)$$

Tractable computation of Bayesian posteriors

Solution: Linear Gaussian Models

- Dynamics (Prediction)

$$\overline{Bel}(x_t) = \int p(x_t | u_t, x_{t-1}) Bel(x_{t-1}) dx_{t-1}$$

- Measurement (Correction)

$$Bel(x_t) = \eta P(z_t | x_t) \overline{Bel}(x_t)$$

Model as Linear Gaussian



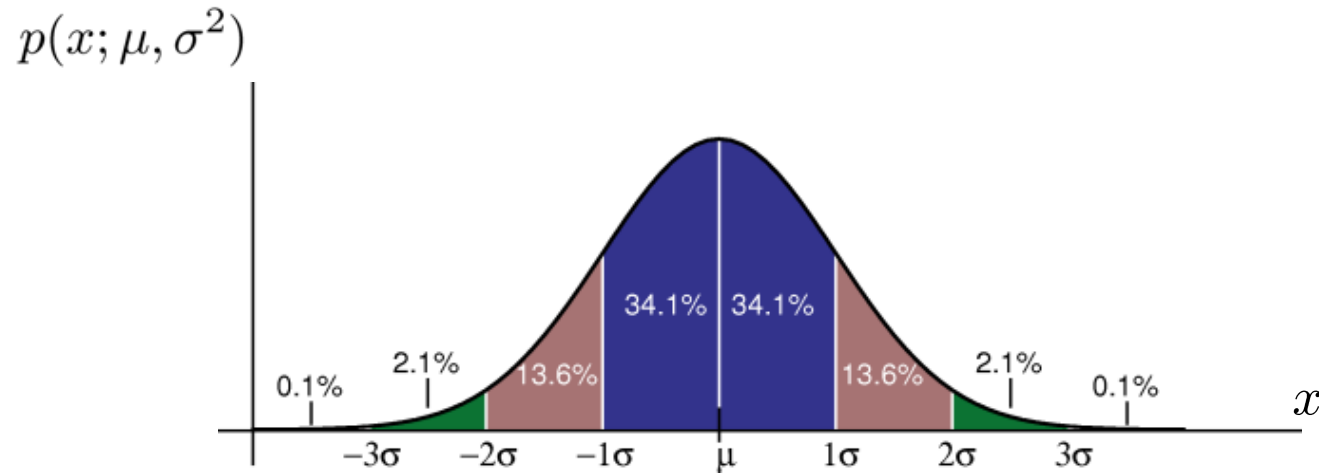
Let's take a little Gaussian detour

Gaussians (1D)

- Gaussian with mean (μ) and standard deviation (σ)

$$X \sim \mathcal{N}(\mu, \sigma^2)$$

$$p(x; \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right)$$



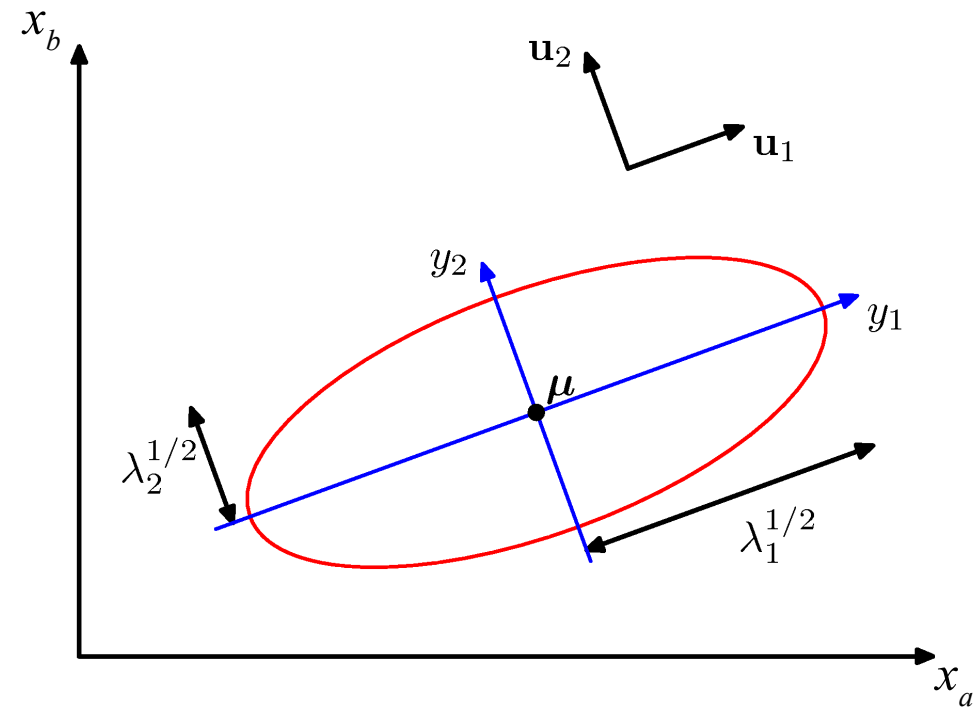
Gaussians (2D) – we won't get too deep into this!

$$p(\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

$$\mathbf{x} = \begin{pmatrix} x_a \\ x_b \end{pmatrix}, \quad \boldsymbol{\mu} = \begin{pmatrix} \mu_a \\ \mu_b \end{pmatrix}$$

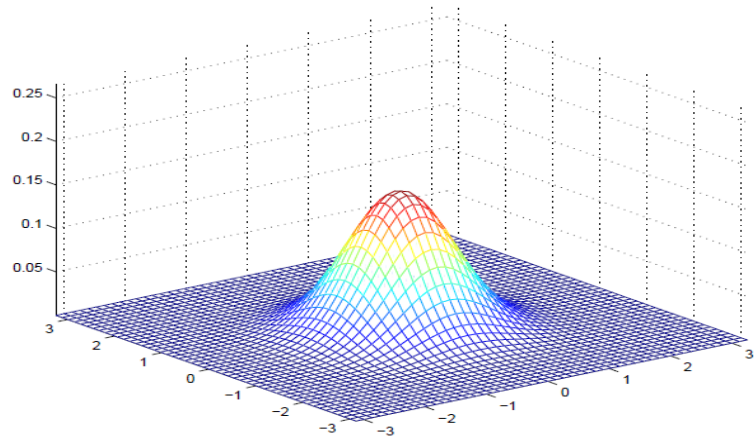
$$\boldsymbol{\Sigma} = \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix}$$

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2} |\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$$

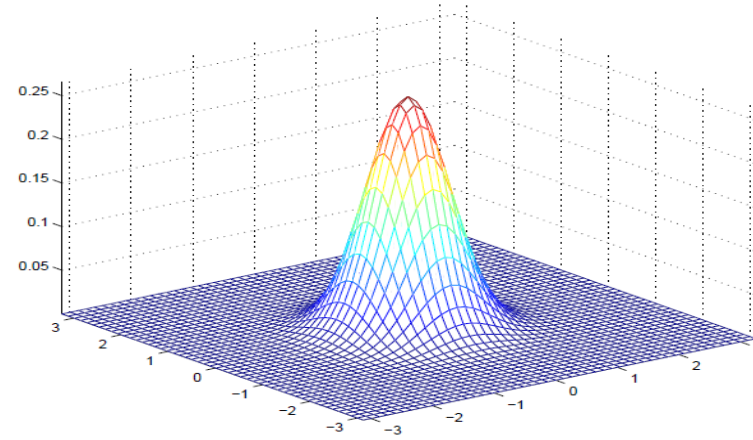


2D examples

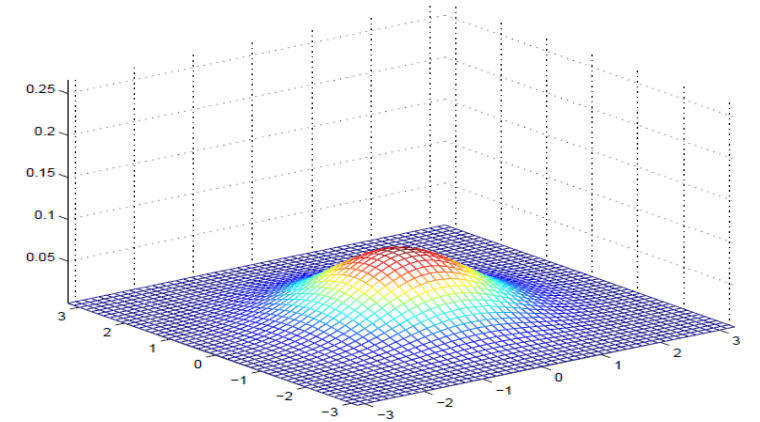
Slide from Pieter Abbeel



- $\mu = [0; 0]$
- $\Sigma = [1 \ 0; 0 \ 1]$



- $\mu = [0; 0]$
- $\Sigma = [.6 \ 0; 0 \ .6]$



- $\mu = [0; 0]$
- $\Sigma = [2 \ 0; 0 \ 2]$

Important Identities: Gaussians

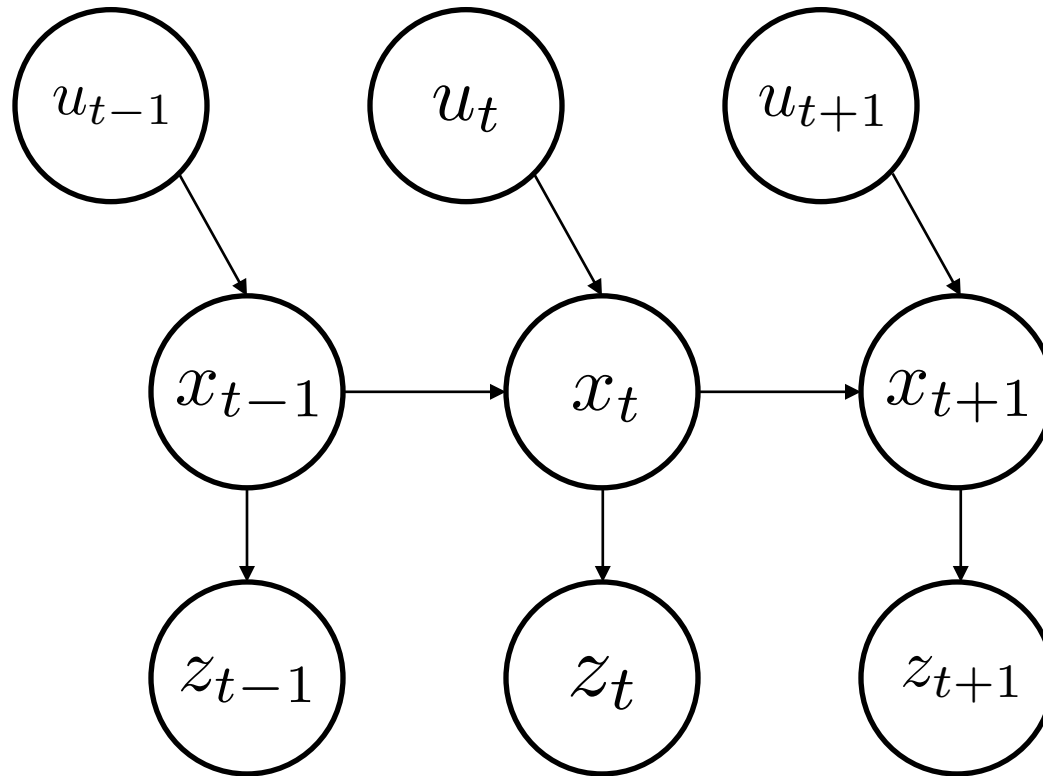
$$\text{Forward propagation} \quad \begin{cases} X \sim \mathcal{N}(\mu, \Sigma) \\ Y = AX + B + \epsilon \\ \epsilon \sim \mathcal{N}(0, Q) \end{cases} \implies Y \sim \mathcal{N}(A\mu + B, A\Sigma A^T + Q)$$

$$\text{Conditioning} \quad \begin{cases} X \sim \mathcal{N}(\mu, \Sigma) \\ Y = CX + B + \delta \\ \delta \sim \mathcal{N}(0, R) \end{cases} \implies X|Y = y_0 \sim \mathcal{N}(\mu + K(y_0 - C\mu), (I - KC)\Sigma)$$

- Marginalization and conditioning in Gaussians results in Gaussians
- We stay in the “Gaussian world” as long as we start with Gaussians and perform only linear transformations.

Discrete Kalman Filter

Kalman filter = Bayes filter with Linear Gaussian dynamics and sensor models



Discrete Kalman Filter: Scalar Version

Estimates the state \mathbf{x} of a discrete-time controlled process that is governed by the linear stochastic difference equation

$$x_t = ax_{t-1} + bu_t + \epsilon_t$$

$$\epsilon_t \sim \mathcal{N}(0, q)$$

with a measurement

$$z_t = cx_t + \delta_t$$

$$\delta_t \sim \mathcal{N}(0, r)$$

Linear Gaussian



Discrete Kalman Filter: Matrix Version

Estimates the state \mathbf{x} of a discrete-time controlled process that is governed by the linear stochastic difference equation

$$\mathbf{x}_t = A\mathbf{x}_{t-1} + B\mathbf{u}_t + \epsilon_t$$

$$\epsilon_t \sim \mathcal{N}(0, Q)$$

with a measurement

$$z_t = C\mathbf{x}_t + \delta_t$$

$$\delta_t \sim \mathcal{N}(0, R)$$

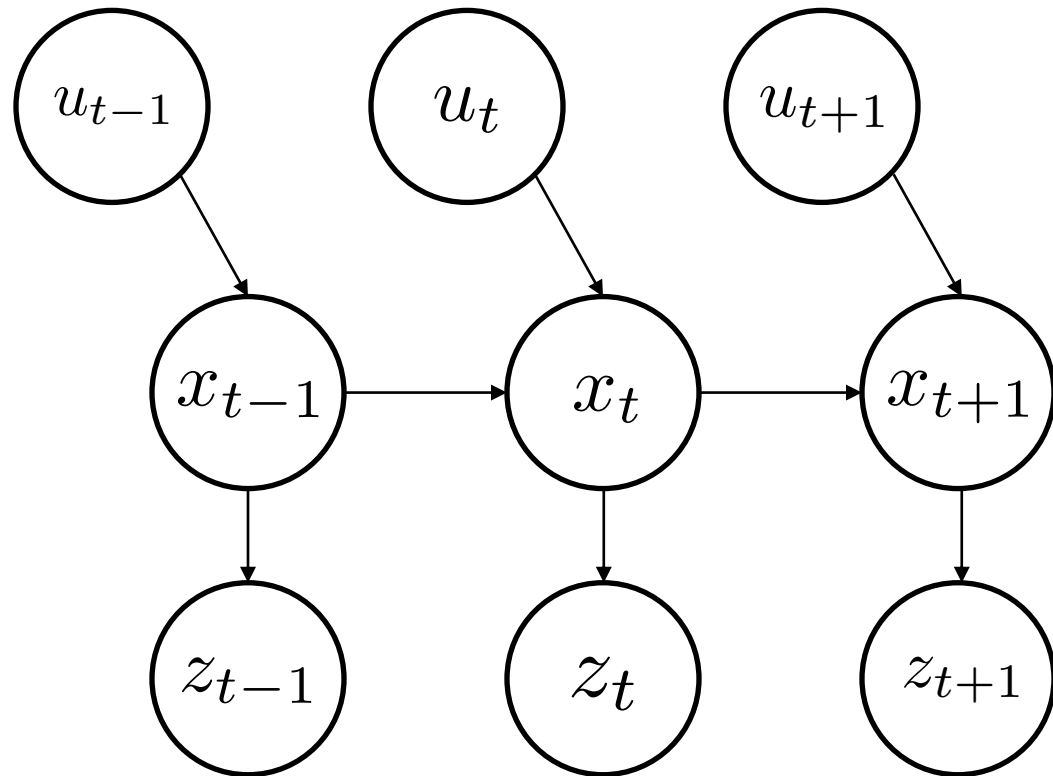
Linear Gaussian



Components of a Kalman Filter

- A Matrix ($n \times n$) that describes how the state evolves from $\mathbf{t-1}$ to \mathbf{t} without controls or noise.
- B Matrix ($n \times l$) that describes how the control $\mathbf{u}_{\mathbf{t-1}}$ changes the state from $\mathbf{t-1}$ to \mathbf{t}
- C Matrix ($k \times n$) that describes how to map the state $\mathbf{x}_{\mathbf{t}}$ to an observation $\mathbf{z}_{\mathbf{t}}$.
- ϵ_t Random variables representing the process and measurement noise that are assumed to be independent and normally distributed with covariance
- δ_t \mathbf{R} and \mathbf{Q} respectively.

Goal of the Kalman Filter: Same as Bayes Filter



Belief

$$p(x_t | z_{0:t}, u_{0:t})$$

Idea: recursive update

$$\propto p(z_t | x_t) \int p(x_t | x_{t-1}, u_t) p(x_{t-1} | z_{0:t-1}, u_{0:t-1})$$

↓ Measurement ↘ Dynamics ↘ Recursive Belief

2 step process:

- Dynamics update (incorporate action)
- Measurement update (incorporate sensor reading)

Bayes Filters

Key Idea: Apply Markov to get a recursive update!

Step 0. Start with the belief at time step $t-1$

$$bel(x_{t-1})$$

Step 1: Prediction - push belief through dynamics given **action**

$$\overline{bel}(x_t) = \sum P(x_t | u_t, x_{t-1}) bel(x_{t-1})$$

Linear Gaussian

Step 2: Correction - apply Bayes rule given **measurement**

$$bel(x_t) = \eta P(z_t | x_t) \overline{bel}(x_t)$$

Linear Gaussian Systems: Initialization

- Initial belief is normally distributed:

$$Bel(x_0) = \mathcal{N}(\mu_0, \Sigma_0)$$

- $Bel(x_t)$ at any step t is: $\mathcal{N}(\mu_{t|0:t}, \Sigma_{t|0:t})$
- $\overline{Bel}(x_t)$ at any step t is: $\mathcal{N}(\mu_{t|0:t-1}, \Sigma_{t|0:t-1})$

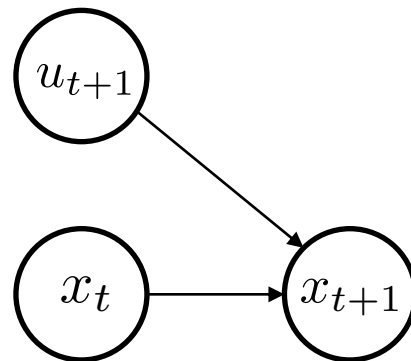
Linear Gaussian Systems: Prediction

- Integrate the effect of one action under the dynamics, before measurement comes in

$$x_{t+1} = Ax_t + Bu_{t+1} + \epsilon_{t+1} \quad \epsilon_{t+1} \sim \mathcal{N}(0, Q_{t+1})$$

$$p(x_{t+1}|x_t, u_{t+1}) = \mathcal{N}(Ax_t + Bu_{t+1}, Q_{t+1})$$

$$\overline{Bel}(x_{t+1}) = \int Bel(x_t) p(x_{t+1}|z_{0:t}, u_{0:t+1}) dx_t$$



Gaussian, easy!

Linear Gaussian Systems: Prediction

- Integrate the effect of one action under the dynamics, before measurement comes in

$$x_{t+1} = Ax_t + Bu_{t+1} + \epsilon_{t+1} \quad \epsilon_{t+1} \sim \mathcal{N}(0, Q_{t+1})$$

$$p(x_{t+1}|x_t, u_{t+1}) = \mathcal{N}(Ax_t + Bu_{t+1}, Q_{t+1})$$

$$\overline{Bel}(x_{t+1}) = \int Bel(x_t) p(x_{t+1}|u_{t+1}, x_t) dx_t$$

$$\begin{cases} X \sim \mathcal{N}(\mu, \Sigma) \\ Y = AX + B + \epsilon \implies Y \sim \mathcal{N}(A\mu + B, A\Sigma A^T + Q) \\ \epsilon \sim \mathcal{N}(0, Q) \end{cases}$$

Gaussian, easy!

Linear Gaussian Systems: Prediction

- Integrate the effect of one action under the dynamics, before measurement comes in

$$p(x_t | u_{0:t}, z_{0:t}) = \mathcal{N}(\mu_{t|0:t}, \Sigma_{t|0:t})$$

$$x_{t+1} = Ax_t + Bu_{t+1} + \epsilon_{t+1}$$

$$\epsilon_{t+1} \sim \mathcal{N}(0, Q_{t+1})$$

$$\begin{cases} X \sim \mathcal{N}(\mu, \Sigma) \\ Y = AX + B + \epsilon \implies Y \sim \mathcal{N}(A\mu + B, A\Sigma A^T + Q) \\ \epsilon \sim \mathcal{N}(0, C) \end{cases}$$

Previous belief

$$p(x_t | u_{0:t}, z_{0:t}) = \mathcal{N}(\mu_{t|0:t}, \Sigma_{t|0:t})$$

Belief Update

$$p(x_{t+1} | u_{0:t+1}, z_{0:t}) = \mathcal{N}(A\mu_{t|0:t} + Bu_{t+1}, A\Sigma_{t|0:t}A^T + Q_{t+1})$$

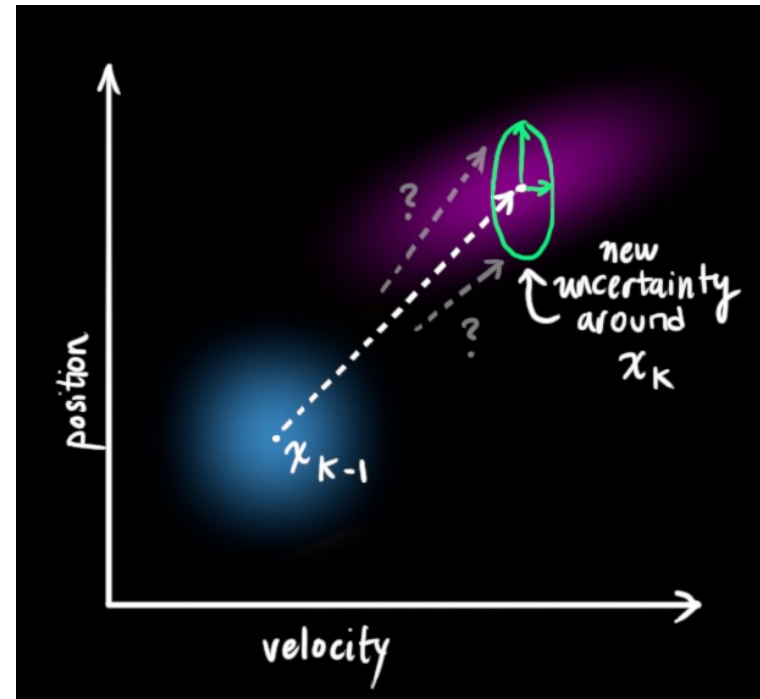
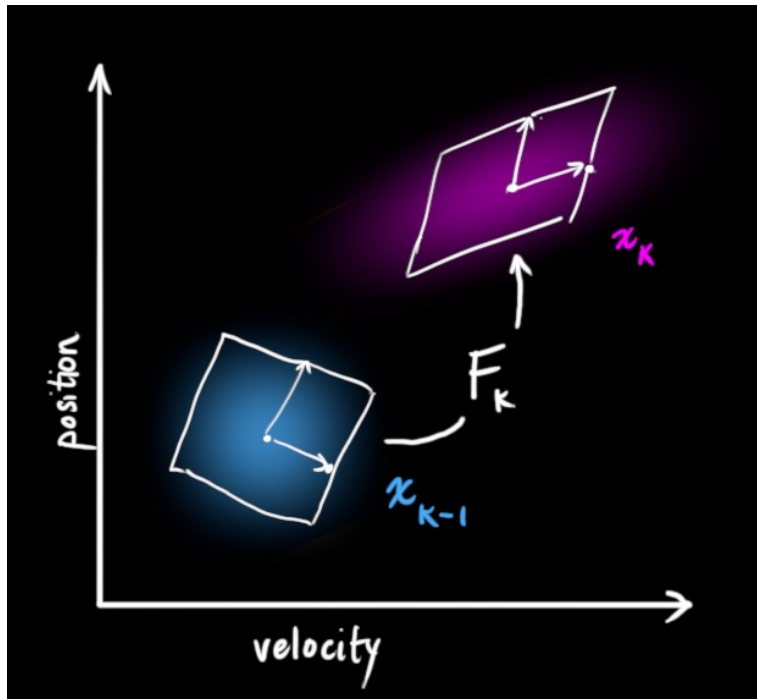
Intuition: Scale and shift the mean according to dynamics, uncertainty grows quadratically!

Linear Gaussian Systems: Prediction

Previous belief $p(x_t | u_{0:t}, z_{0:t}) = \mathcal{N}(\mu_{t|0:t}, \Sigma_{t|0:t})$

Belief Update $p(x_{t+1} | u_{0:t+1}, z_{0:t}) = \mathcal{N}(A\mu_{t|0:t} + Bu_{t+1}, A\Sigma_{t|0:t}A^T + Q_{t+1})$

Intuition: Scale and shift the mean according to dynamics, uncertainty grows!



Intuition Behind Prediction Step

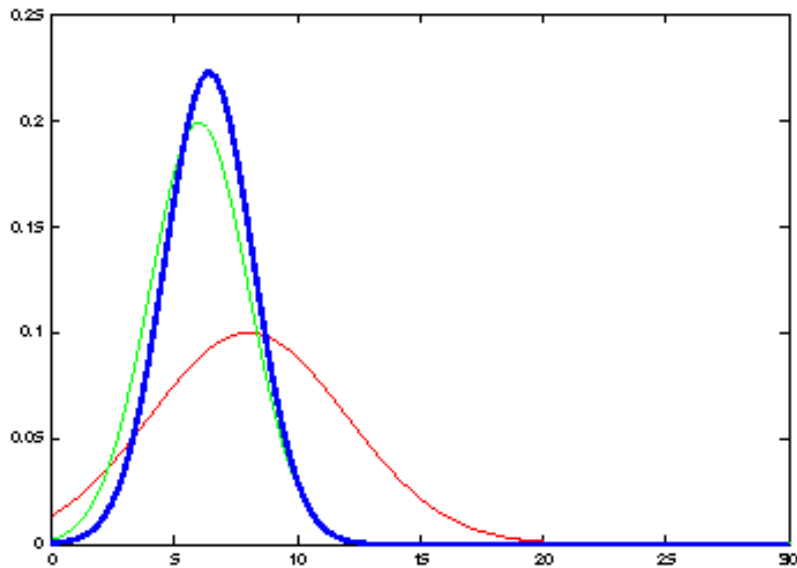
Previous belief

$$p(x_t | u_{0:t}, z_{0:t}) = \mathcal{N}(\mu_{t|0:t}, \Sigma_{t|0:t})$$

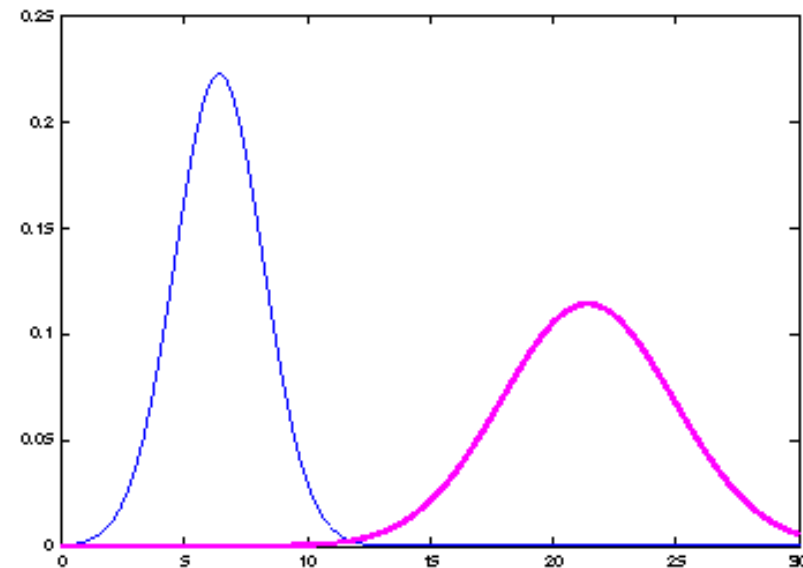
Belief Update

$$p(x_{t+1} | u_{0:t+1}, z_{0:t}) = \mathcal{N}(A\mu_{t|0:t} + Bu_{t+1}, A\Sigma_{t|0:t}A^T + Q_{t+1})$$

Intuition: Scale and shift the mean according to dynamics, uncertainty grows!



Belief at x_t



Belief post dynamics \rightarrow shifted mean, scaled and shifted variance

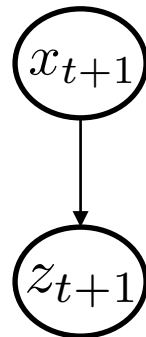
Linear Gaussian Systems: Observations

- Integrate the effect of an observation using sensor model, after dynamics

$$z_{t+1} = Cx_{t+1} + \delta_{t+1} \quad \delta_{t+1} \sim \mathcal{N}(0, R_{t+1})$$

$$p(z_{t+1}|x_{t+1}) = \mathcal{N}(Cx_{t+1}, R_{t+1})$$

$$p(x_{t+1}|u_{0:t+1}, z_{0:t+1}) \propto \overset{Bel(x_{t+1})}{p(z_{t+1}|x_{t+1})} \overset{\overline{Bel}(x_{t+1})}{p(x_{t+1}|u_{0:t+1}, z_{0:t})}$$



Gaussian, easy to normalize

Slightly harder than the dynamics step!

Linear Gaussian Systems: Observations

- Integrate the effect of an observation using sensor model, after dynamics

$$z_{t+1} = Cx_{t+1} + \delta_{t+1} \quad \delta_{t+1} \sim \mathcal{N}(0, R_{t+1})$$

$$p(z_{t+1}|x_{t+1}) = \mathcal{N}(Cx_{t+1}, R_{t+1})$$

$$p(x_{t+1}|u_{0:t+1}, z_{0:t+1}) \propto \overset{Bel(x_{t+1})}{p(x_{t+1}|u_{0:t+1}, z_{0:t})} \overset{\overline{Bel}(x_{t+1})}{p(z_{t+1}|x_{t+1})}$$

$$\text{Conditioning} \quad \begin{cases} X \sim \mathcal{N}(\mu, \Sigma) \\ Y = CX + B + \delta \\ \delta \sim \mathcal{N}(0, R) \end{cases} \implies X|Y = y_0 \sim \mathcal{N}(\mu + K(y_0 - C\mu), (I - KC)\Sigma)$$
$$K = \Sigma C^T (C\Sigma C^T + R)^{-1}$$

Linear Gaussian Systems: Observations

- Integrate the effect of an observation using sensor model, after dynamics

$$p(x_{t+1}|u_{0:t+1}, z_{0:t}) = \mathcal{N}(\mu_{t+1|0:t}, \Sigma_{t+1|0:t})$$

$$z_{t+1} = Cx_{t+1} + \delta_{t+1}$$

$$\delta_{t+1} \sim \mathcal{N}(0, R_{t+1})$$

$$\begin{cases} X \sim \mathcal{N}(\mu, \Sigma) \\ Y = CX + B + \delta \\ \delta \sim \mathcal{N}(0, R) \end{cases} \implies X|Y = y_0 \sim \mathcal{N}(\mu + K(y_0 - C\mu), (I - KC)\Sigma)$$
$$K = \Sigma C^T (C\Sigma C^T + R)^{-1}$$

Previous belief $p(x_{t+1}|u_{0:t+1}, z_{0:t}) = \mathcal{N}(\mu_{t+1|0:t}, \Sigma_{t+1|0:t})$ Computed from dynamics step

Updated belief $p(x_{t+1}|u_{0:t+1}, z_{0:t+1})$
 $= \mathcal{N}(\mu_{t+1|0:t} + K_{t+1}(z_{t+1} - C\mu_{t+1|0:t}), (I - K_{t+1}C)\Sigma_{t+1|0:t})$

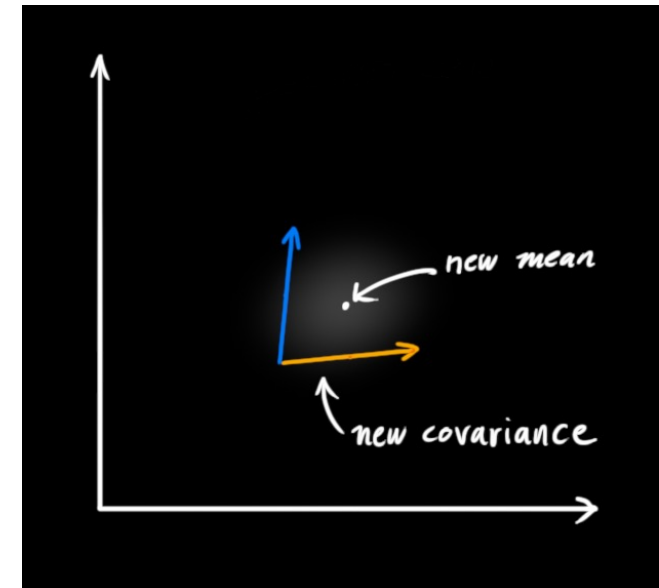
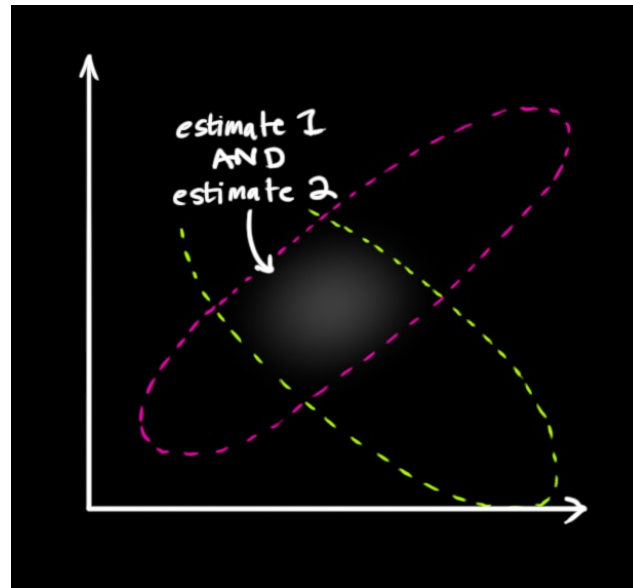
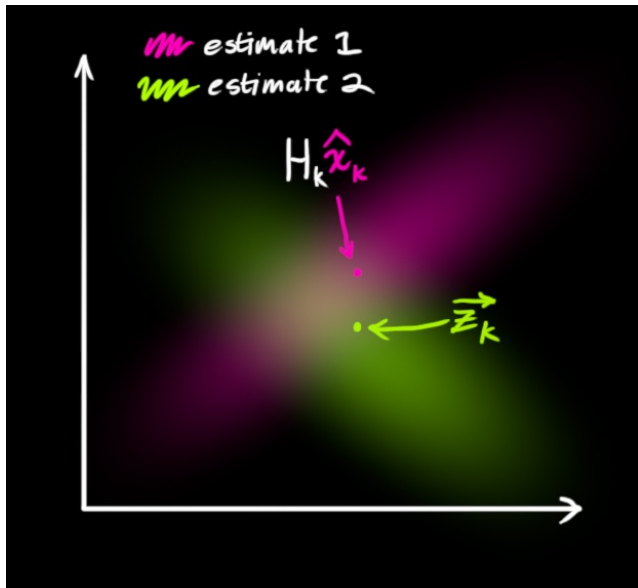
$$K_{t+1} = \Sigma_{t+1|0:t} C^T (C\Sigma_{t+1|0:t} C^T + R)^{-1}$$

Linear Gaussian Systems: Observations

Previous belief $p(x_{t+1} | u_{0:t+1}, z_{0:t}) = \mathcal{N}(\mu_{t+1|0:t}, \Sigma_{t+1|0:t})$ Computed from dynamics step

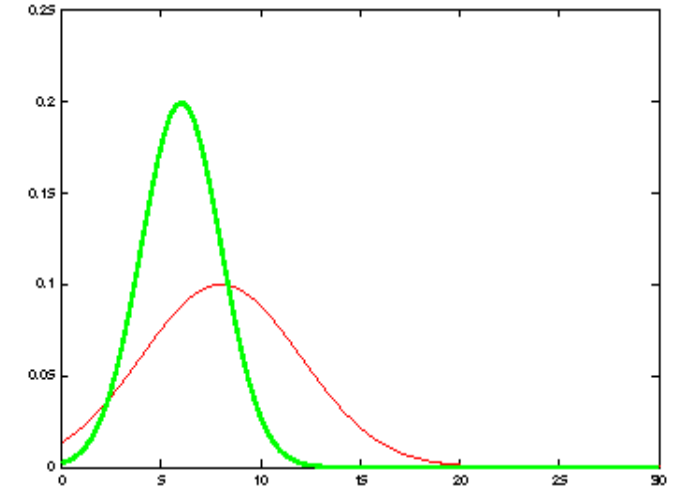
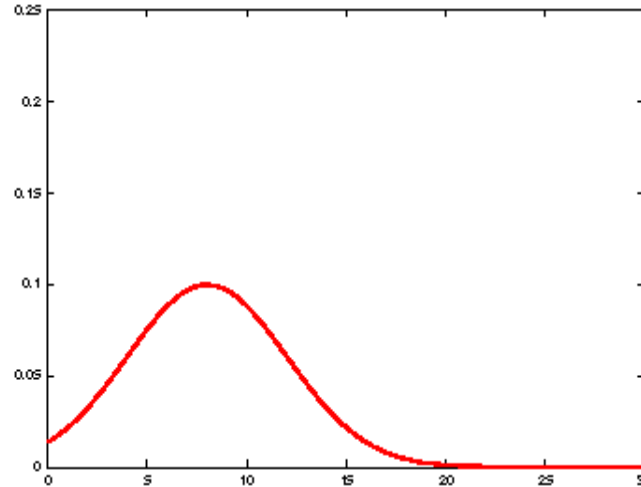
Updated belief $p(x_{t+1} | u_{0:t+1}, z_{0:t+1})$
 $= \mathcal{N}(\mu_{t+1|0:t} + K_{t+1}(z_{t+1} - C\mu_{t+1|0:t}), (I - K_{t+1}C)\Sigma_{t+1|0:t})$

Intuition: Correct the update linearly according to measurement error from expectation, shrink uncertainty accordingly



Intuition Behind Correction Step

- Previous belief
- New Measurement



$$p(x_{t+1}|u_{0:t+1}, z_{0:t+1}) = \mathcal{N}(\mu_{t+1|0:t} + K_{t+1}(z_{t+1} - C\mu_{t+1|0:t}), (I - K_{t+1}C)\Sigma_{t+1|0:t})$$

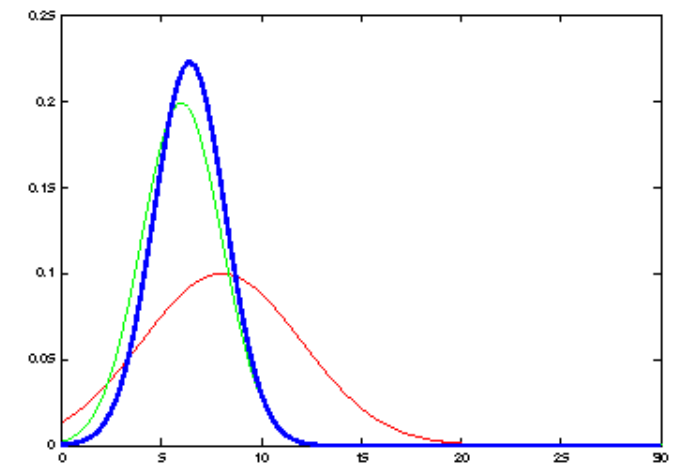
$$K_{t+1} = \Sigma_{t+1|0:t}C^T(C\Sigma_{t+1|0:t}C^T + R)^{-1}$$

For the sake of simplicity, let's say $C = I$

$$K_{t+1} = \frac{\Sigma_{t+1|0:t}}{\Sigma_{t+1|0:t} + R}$$

Corrects belief based on measurement

- Average between mean and measurement based on K
- Scale down uncertainty based on K



Unpacking the Kalman Gain

Previous belief $p(x_{t+1} | u_{0:t+1}, z_{0:t}) = \mathcal{N}(\mu_{t+1|0:t}, \Sigma_{t+1|0:t})$ Computed from dynamics step

Updated belief $p(x_{t+1} | u_{0:t+1}, z_{0:t+1})$
 $= \mathcal{N}(\mu_{t+1|0:t} + K_{t+1}(z_{t+1} - C\mu_{t+1|0:t}), (I - K_{t+1}C)\Sigma_{t+1|0:t})$

$$K_{t+1} = \Sigma_{t+1|0:t} C^T (C \Sigma_{t+1|0:t} C^T + R)^{-1}$$

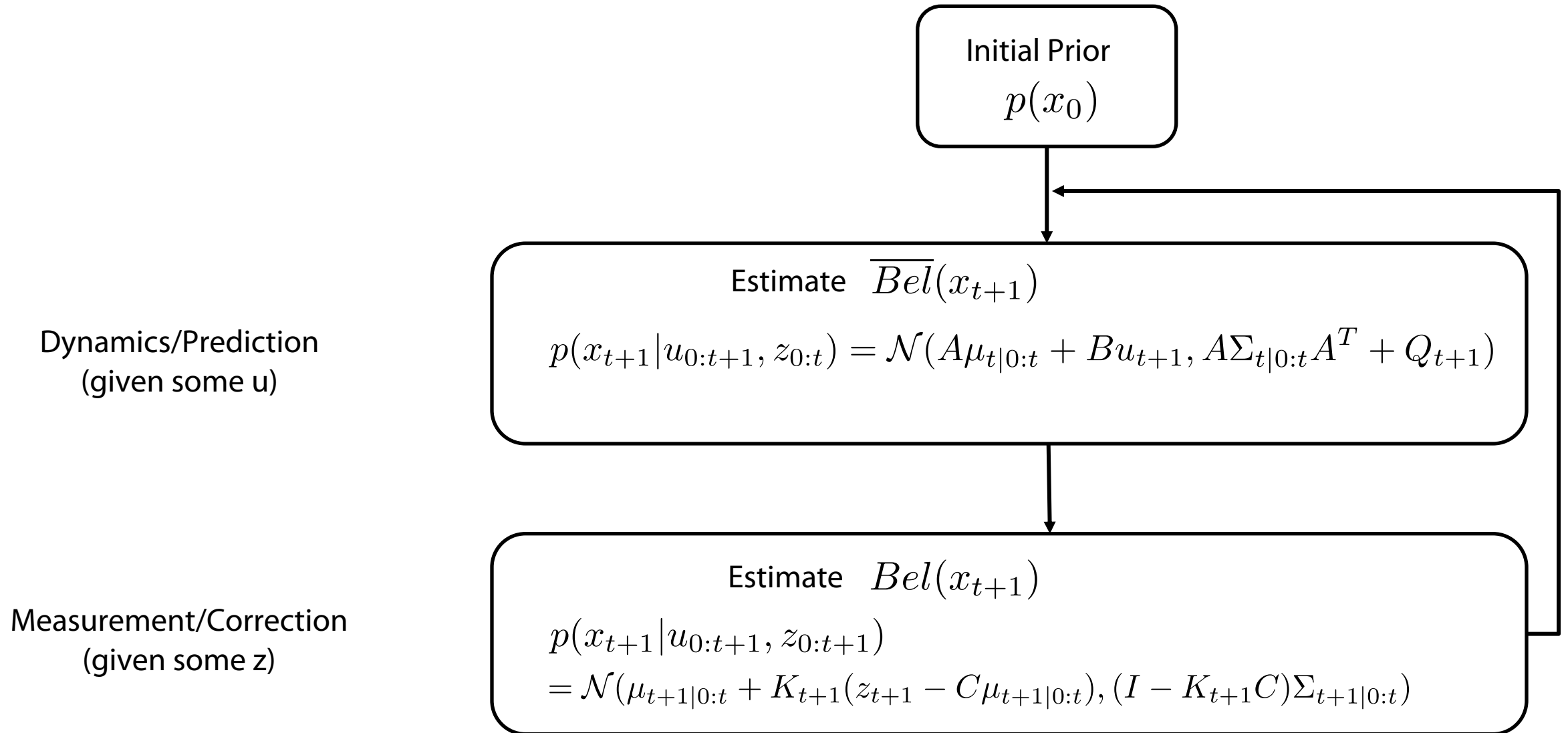
Case 1: Very noisy sensor, $R \gg \Sigma$

For the sake of simplicity, let's say $C = I$

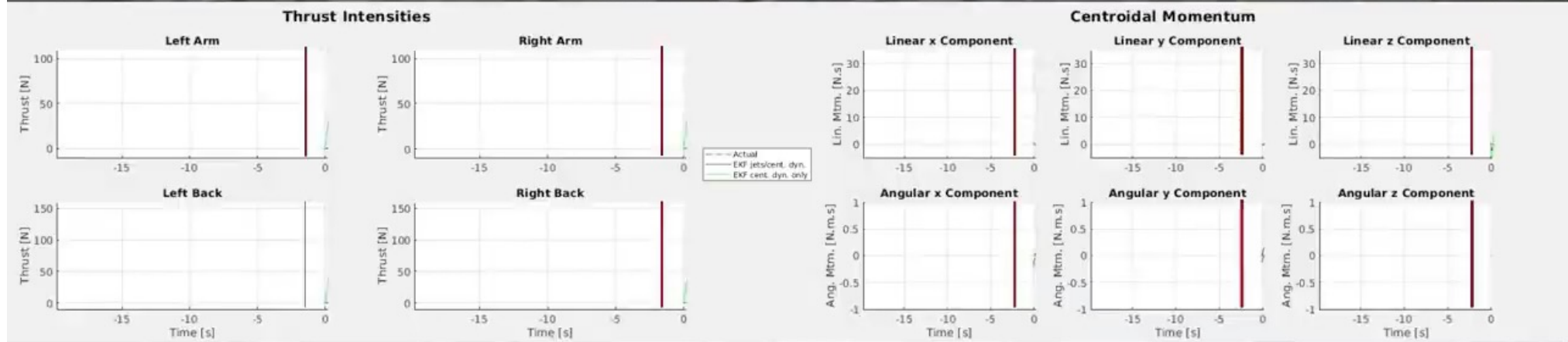
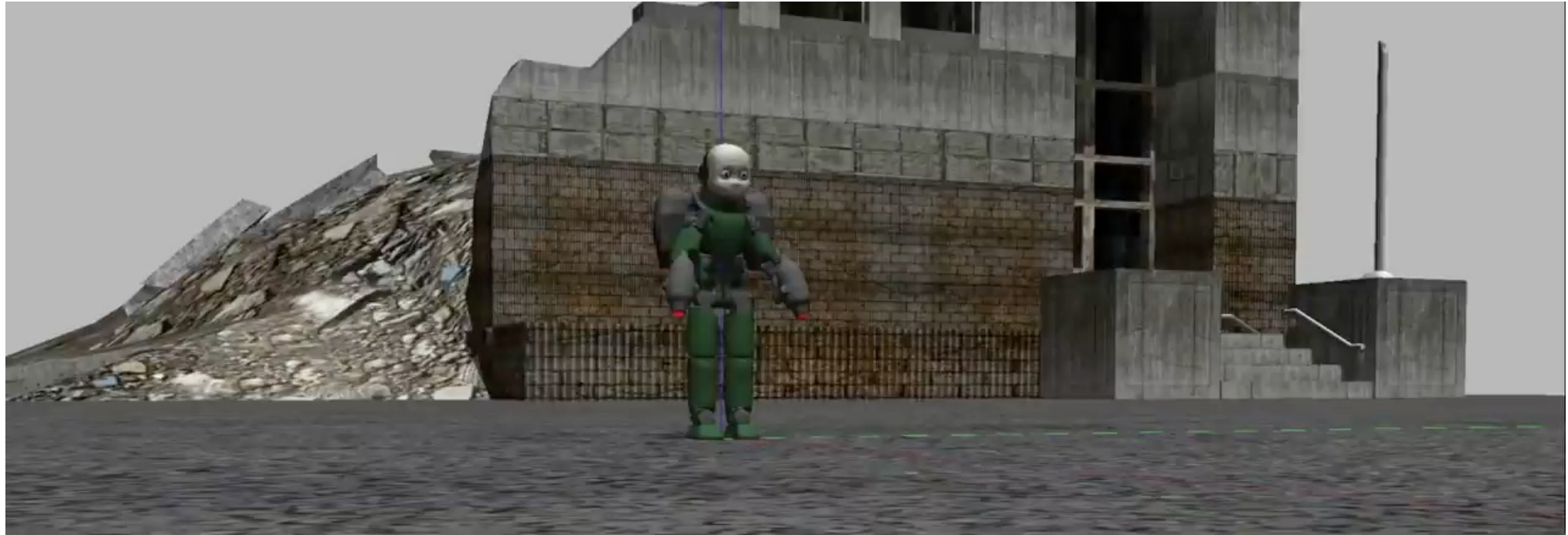
$$K_{t+1} = \frac{\Sigma_{t+1|0:t}}{\Sigma_{t+1|0:t} + R}$$

Case 2: Deterministic sensor, $R = 0$

Kalman Filter Algorithm



Kalman Filter in Action



Kalman Filter Summary

- **Highly efficient:** Polynomial in measurement dimensionality k and state dimensionality n :
 $O(k^{2.376} + n^2)$

Matrix Inversion (Correction)

$$K_{t+1} = \Sigma_{t+1|0:t} C^T (C \Sigma_{t+1|0:t} C^T + R_{t+1})^{-1}$$

Matrix Multiplication (Prediction)

$$p(x_{t+1}|z_{0:t}, u_{0:t+1}) \sim \mathcal{N}(A\mu_{t|0:t} + Bu_t, A\Sigma_{t|0:t}A^T + Q_t)$$

- **Optimal for linear Gaussian systems!**
- **Most robotics systems are nonlinear!**

Why should we care?

Still a very widely used technique for estimation/localization/mapping in real problems

