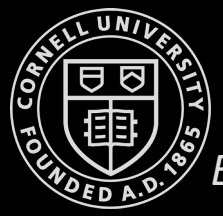
ECE 4960

Prof. Kirstin Hagelskjær Petersen
kirstin@cornell.edu

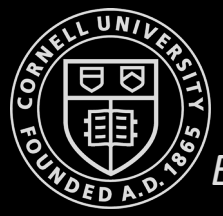
Fast Robots



Linear Systems Control

Based on “Control Bootcamp”, Steve Brunton, UW
https://www.youtube.com/watch?v=Pi7l8mMjYVE

- Linear systems review
- Eigenvectors and eigenvalues
- Stability
- Discrete time systems
- Linearizing non-linear systems
- Controllability
  - LQR
- Observability



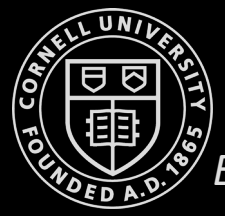
Linear Systems Control – "review of review"

- Linear system: $\dot{x} = Ax$
- Solution: $x(t) = e^{At}x(0)$
- Eigenvectors: $T = [\xi_1 \quad \xi_2 \quad \dots \quad \xi_n]$
- Eigenvalues: $\mathrm{D} = \begin{bmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \dots & \\ 0 & & & \lambda_n \end{bmatrix}$

```
>>[T,D] = eig(A)
```

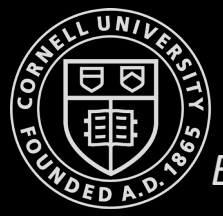
- Linear transform: $AT = TD$
- Solution: $e^{At} = Te^{Dt}T^{-1}$
- Mapping from z to x: $x(t) = Te^{Dt}T^{-1}x(0)$
- Stability in continuous time: $\lambda = a + ib$, stable iff a<0
  - Discrete time: $x(k+1) = \tilde{A}x(k), \tilde{A} = e^{A\Delta t}$
    - Stability in discrete time: $\tilde{\lambda}^n = R^n e^{in\theta}$, stable iff R<1

- Linearizing non-linear systems
  - Fixed points
  - Jacobian
- Controllability
  - `>>rank(ctrb(A,B))`
- Reachability
- Controllability Gramian
- Pole placement
  - $\dot{x} = (A - BK)x$

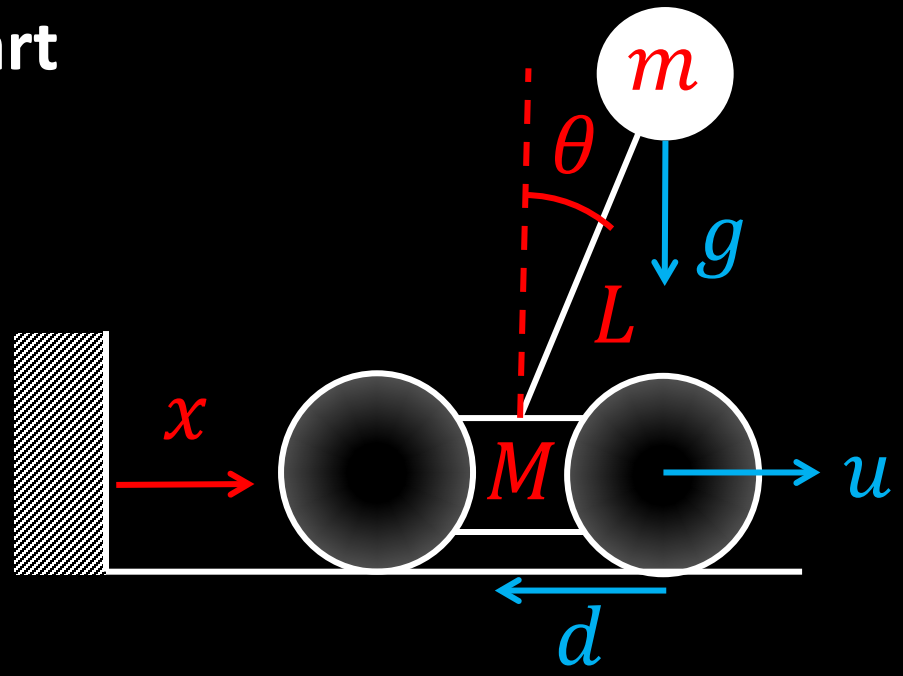


Prof. Kirstin Hagelskjær Petersen
kirstin@cornell.edu

Inverted Pendulum on a Cart



Inverted Pendulum on a Cart



Force acting on the cart in the x direction

Eq. of motion

↓

State space model

$$\underline{x} = \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix}$$

⟶ Fixed points, $\bar{x}$

down

$$\theta = 0, \pi$$

up

$$\dot{\theta} = 0$$

$$\dot{x} = 0$$

x free variable

⟶ Jacobian

$$\frac{df}{dx}\Big|_{\bar{x}}$$

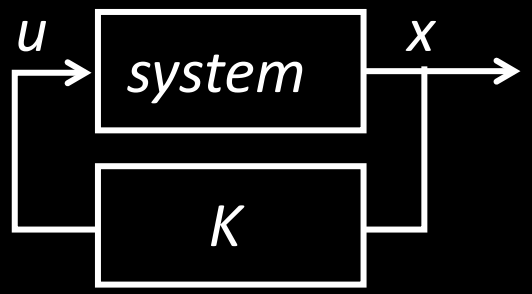
$$\dot{x} = Ax + Bu$$

⟶ (A,B) Controllable?

↓

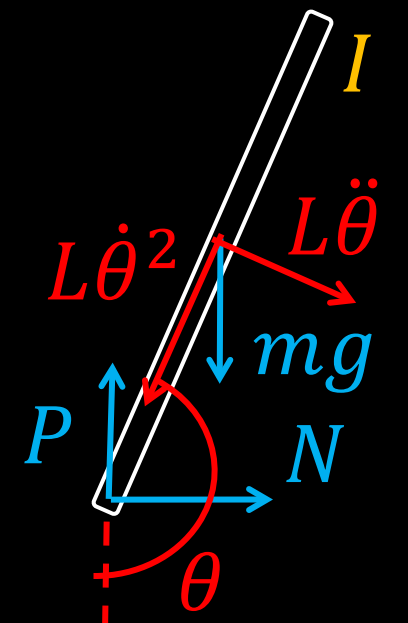
Add linear control

$$\dot{x} = (A - BK)x$$

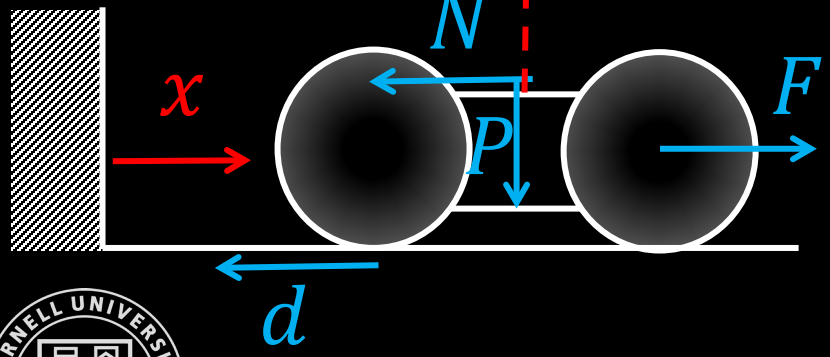


Inverted Pendulum on a Cart – State Space

Free-body diagrams



$$\underline{x} = \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix}$$



Linearized about $\theta = \pi$

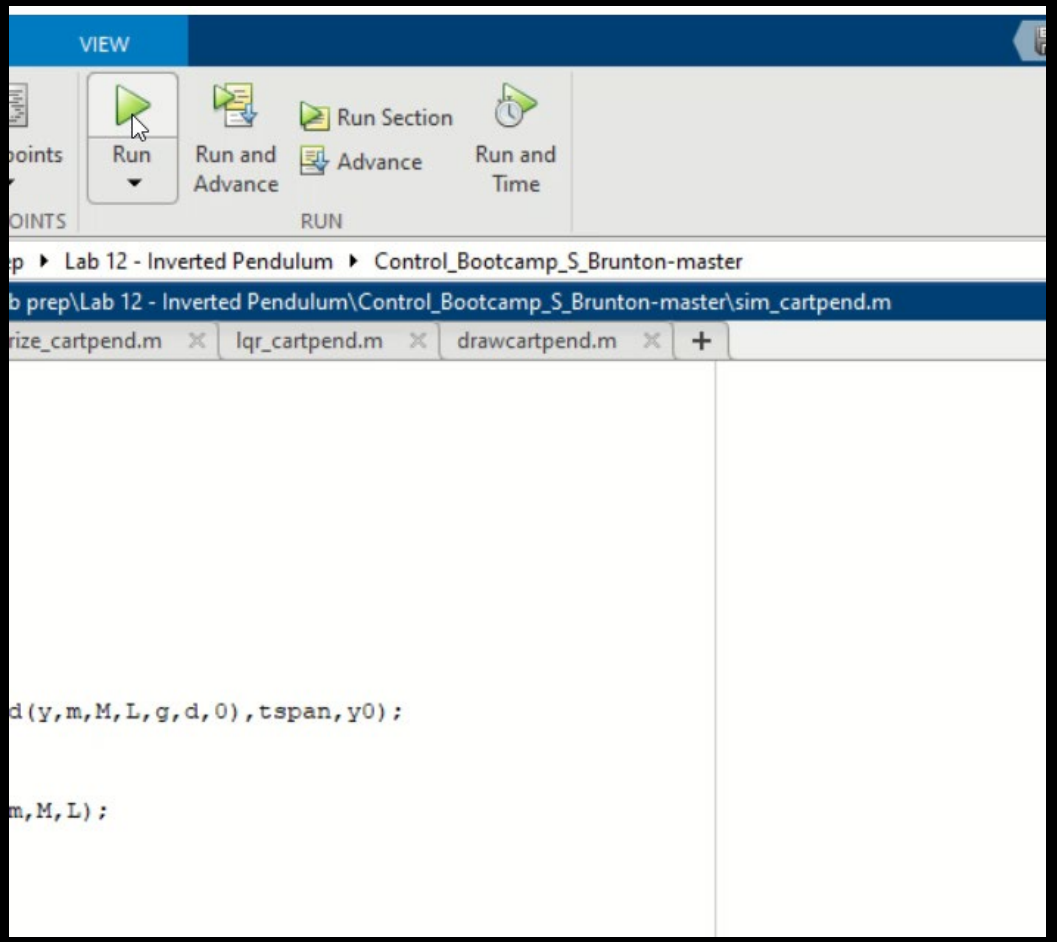
- (14) $\ddot{\varphi} = \frac{(M+m)g}{ML}\varphi - \frac{d}{ML}\dot{x} + \frac{1}{ML}u$
- (16) $\ddot{x} = \frac{m}{M}g\varphi - \frac{d}{M}\dot{x} + \frac{1}{M}u$
- $$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{d}{M} & \frac{m}{M}g & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{d}{ML} & \frac{(M+m)g}{ML} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{ML} \end{bmatrix} u$$

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-(I+ml^2)b}{I(M+m)+Mml^2} & \frac{m^2gl^2}{I(M+m)+Mml^2} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-mlb}{I(M+m)+Mml^2} & \frac{mgl(M+m)}{I(M+m)+Mml^2} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{(I+ml^2)b}{I(M+m)+Mml^2} \\ 0 \\ \frac{-ml}{I(M+m)+Mml^2} \end{bmatrix} u$$

Inverted Pendulum on a Cart

Matlab example

- Non-linear model
- Linearized model
- Eigenvalues
- Stability
- Controllability



$$\gg eig(A)$$

$$\lambda_4 = 3.5069$$

$$\lambda_3 = -1.9278$$

$$\lambda_2 = -3.6844$$

$$\lambda_1 = 0$$

$$\gg rank(ctrb(A, B))$$

$$4$$

Inverted Pendulum on a Cart

Matlab example
- Non-linear model
- Linearized model
- Eigenvalues
- Stability
- Controllability
- Add control

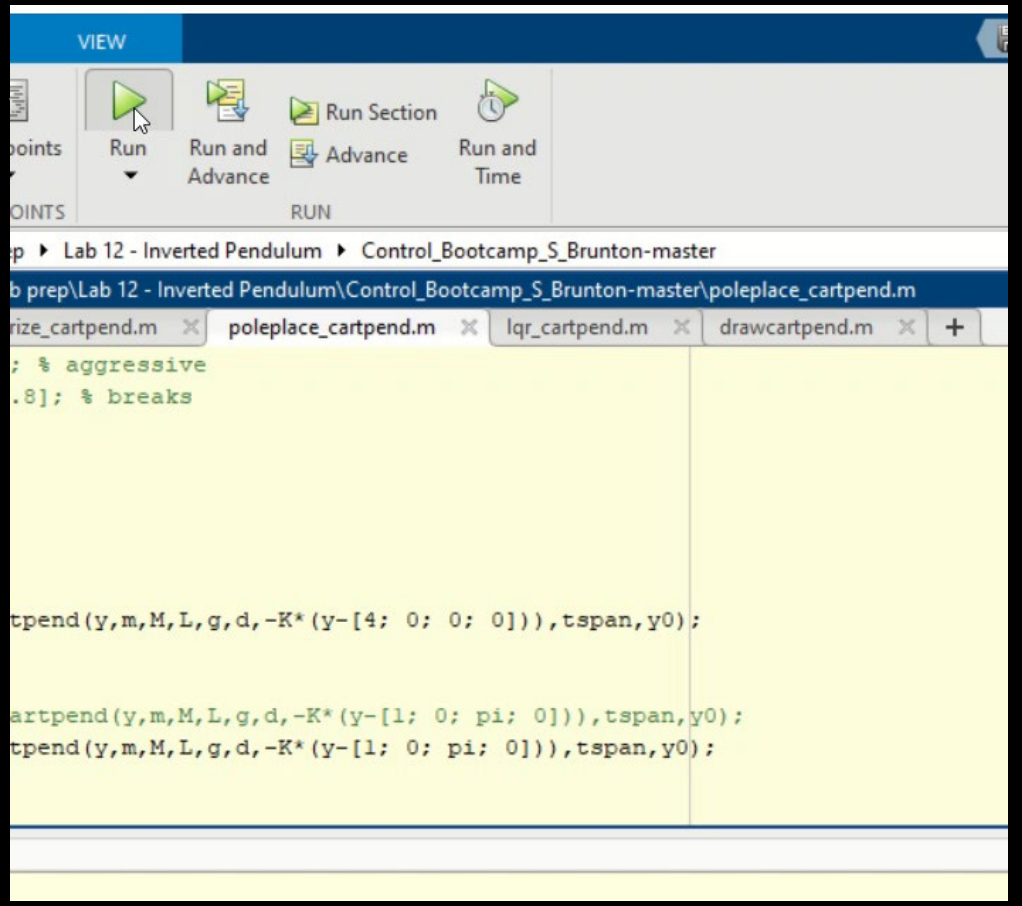
$\gg eigs = [-1.1; -1.2; -1.3; -1.4]$

$\gg K = place(A,B,eigs)$

$K=[-0.0965 \quad -1.3111 \quad 8.7254 \quad 2.2295]$

$\gg eig(A-B.*K)$

$[-1.4; -1.3; -1.2; -1.$



Pole Placement

- In Python
  - https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.place_poles.html
  - K = scipy.signal.place_poles(A, B, poles)
- Barely stable eigenvalues
  - Not enough control authority
- More negative eigenvalues
  - Faster dynamics
  - Less robust system
- Linear Quadratic Control (LQR)
  - “Sweet spot of eigenvalues”
  - Balances how fast you stabilize your state and how much control energy you spend to get there

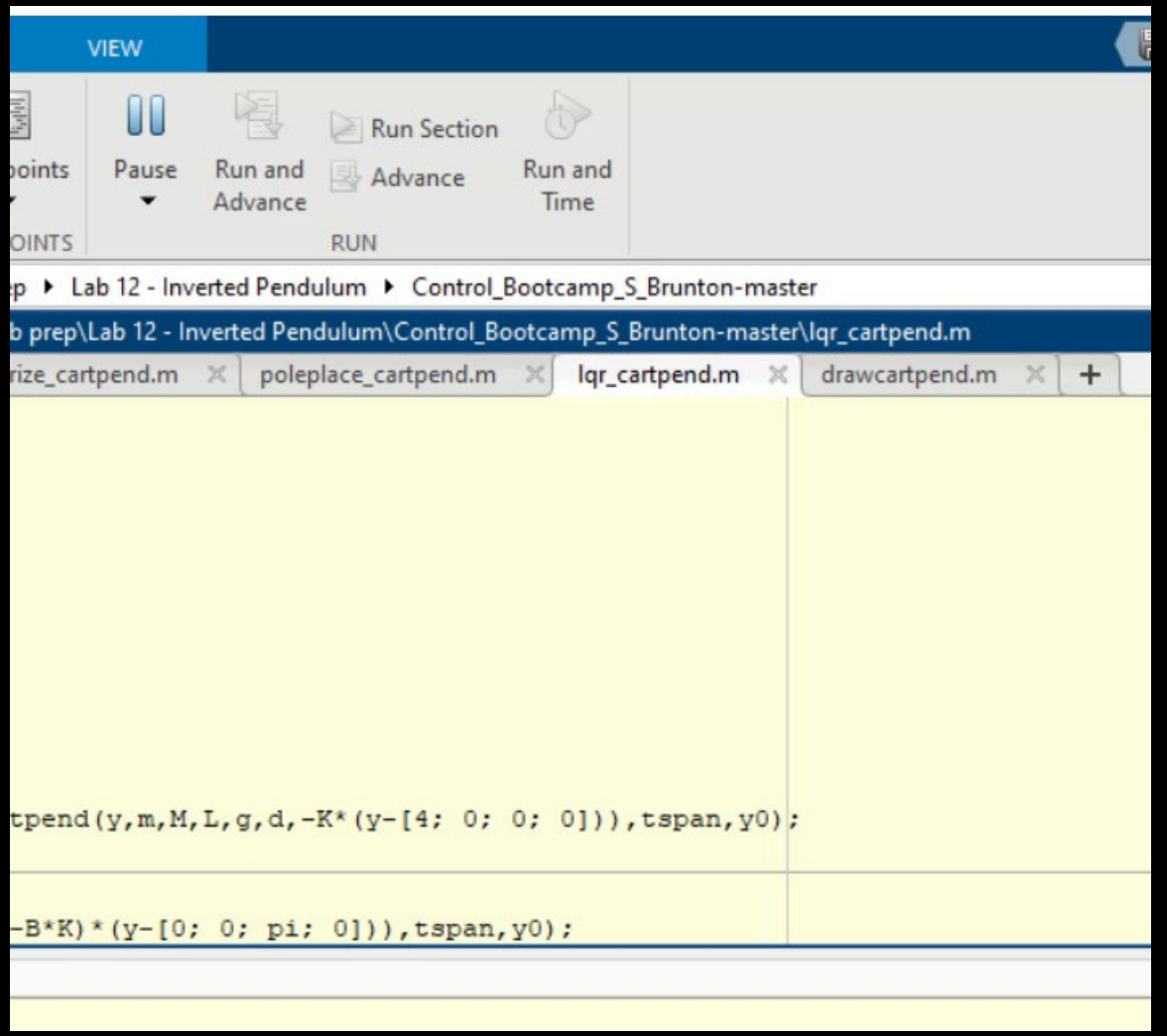
Linear Quadratic Control

$$\dot{x} = Ax + Bu, x \epsilon \mathbb{R}^n$$
$$u = -Kx$$
$$\dot{x} = (A - BK)x$$

- >> K = place(A,B,eigs)
- Where are the best eigs??
  - Linear Quadratic Regulator (LQR)
    - >> K = lqr(A,B,Q,R)
    - $Q = \begin{bmatrix} 1 & & & 0 \\ & 1 & & \\ & & 10 & \\ 0 & & & 100 \end{bmatrix}, R = 0.001$
    - Ricotta equation
      - $\int_0^\infty (x^T Q x + u^T R u) dt$
      - Computationally expensive, O(n^3)

Matlab Example

- $Q = \begin{bmatrix} 1 & & & 0 \\ & 1 & & \\ & & 10 & \\ 0 & & & 100 \end{bmatrix}, R = 0.001$
- K = lqr(A,B,Q,R);
- >>[T,D] = eigs(A-B.*K)
  - λ_1= -788.29 + 0.00i
  - λ_2= -0.70 + 0.83i
  - λ_3=-0.70 - 0.83i
  - λ_4=-0.83 + 0.00i
- T(:,1)
  - = [0.0008, -0.6387, 0.0010, -0.7695]^T

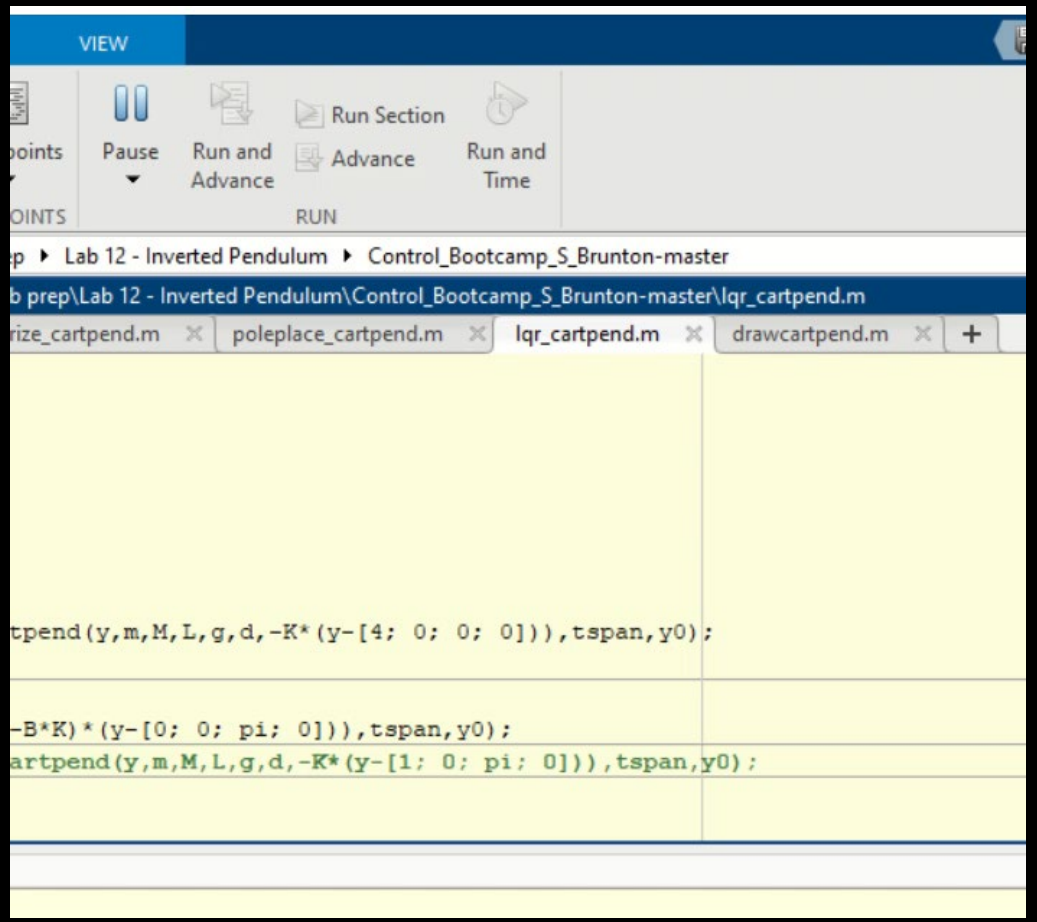


Matlab Example

- $Q = \begin{bmatrix} 1 & & & 0 \\ & 1 & & \\ & & 10 & \\ 0 & & & 100 \end{bmatrix}, R = 0.001$
- K = lqr(A,B,Q,R);
- >>[T,D] = eigs(A-B.*K)
  - λ_1= -788.29 + 0.00i
  - λ_2= -0.70 + 0.83i
  - λ_3=-0.70 - 0.83i
  - λ_4=-0.83 + 0.00i
- T(:,1)
  - = [0.0008, -0.6387, 0.0010, -0.7695]^T

$R = 1$

λ_1 = -25.6851 + 0.0000i
λ_2 = -1.0855 + 0.8921i
λ_3 = -1.0855 - 0.8921i
λ_4 = -0.4811 + 0.0000i

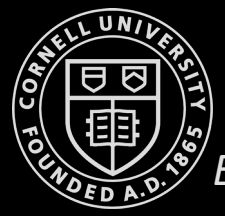


Linear Quadratic Control

$$\dot{x} = Ax + Bu, x \epsilon \mathbb{R}^n$$
$$u = -Kx$$
$$\dot{x} = (A - BK)x$$

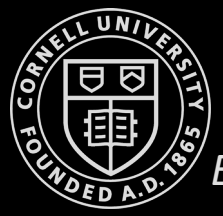
- >> K = place(A,B,eigs)
- Where are the best eigs??
  - Linear Quadratic Regulator (LQR)
    - >> K = lqr(A,B,Q,R)
    - $\int_0^\infty (x^T Q x + u^T R u) dt$
    - $Q = \begin{bmatrix} 1 & & 0 \\ & 1 & \\ & & 10 \\ 0 & & & 100 \end{bmatrix}, R = 0.00$
    - Riccati equation
      - Computationally expensive, O(n^3)

- *The linear controller works!*
  - *(in simulation)*
- *Issues in Practice?*
  - *Imperfect models*
  - *Nonlinear parts*
    - *Deadband, saturation, etc.*
  - *We don't have full state feedback*



Prof. Kirstin Hagelskjær Petersen
kirstin@cornell.edu

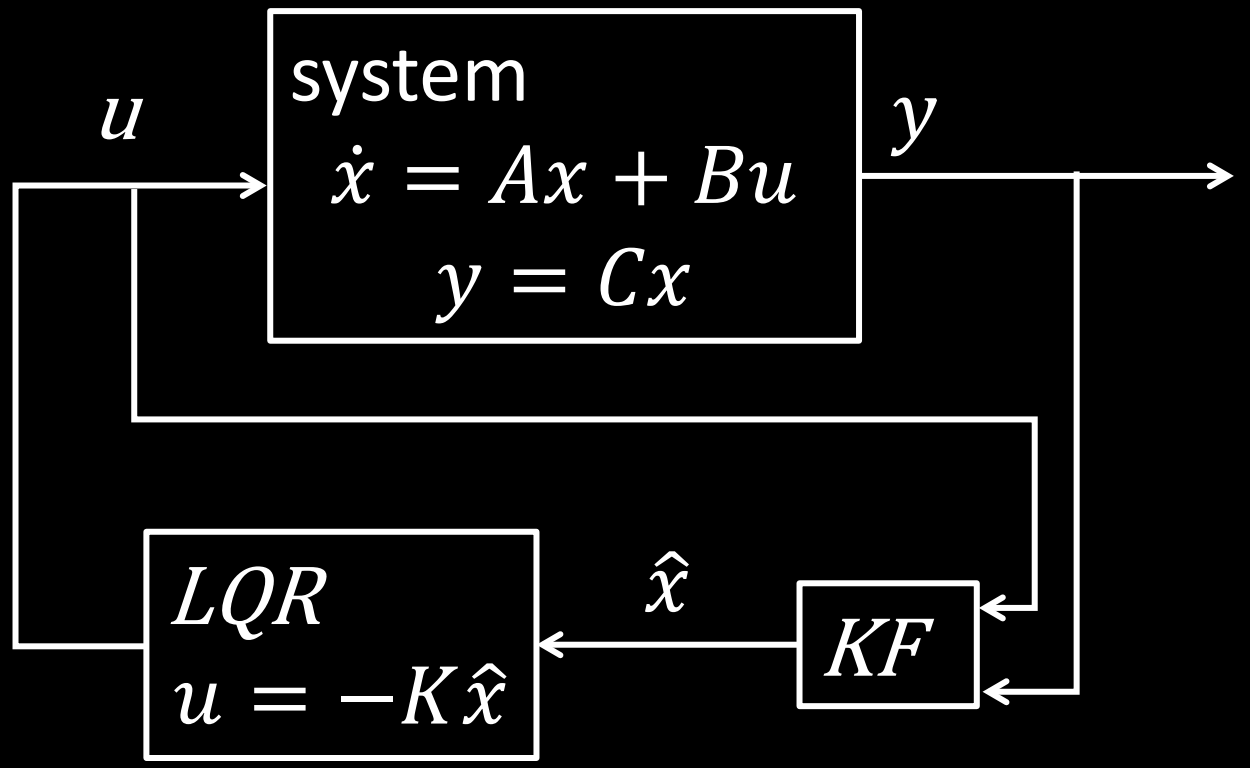
Observability



Observability

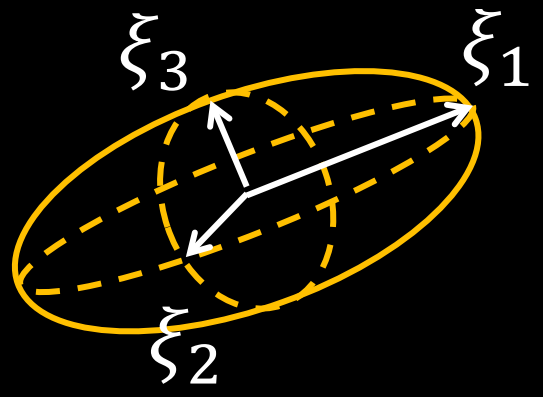
- Controllability
  - Can we steer the system anywhere given some control input u?
- Observability
  - Can we estimate any state x, from a time series of your measurements y(t)?

$$\dot{x} = Ax + Bu, x \epsilon \mathbb{R}^n$$
$$u = -Kx$$
$$\dot{x} = (A - BK)x$$



Observability

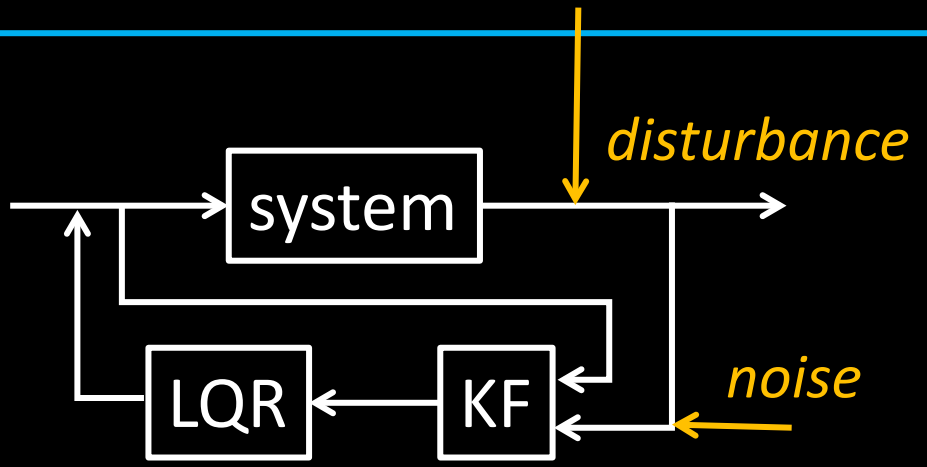
- $\mathcal{O} = \begin{bmatrix} C \\ CA \\ CA^2 \\ \dots \\ CA^{n-1} \end{bmatrix}$



1. Observable iff rank($\mathcal{O}$) = n
   - `>>rank(obsv(A,C))`
2. Iff a system Is observable, we can estimate x from y
   - Observability Gramian
     - `>>[U,Σ, V]=svd(`$\mathcal{O}$`)`

$$\dot{x} = Ax + Bu + d \qquad x \epsilon \mathbb{R}^n$$
$$y = Cx + n \qquad u \epsilon \mathbb{R}^q$$
$$y \epsilon \mathbb{R}^p$$

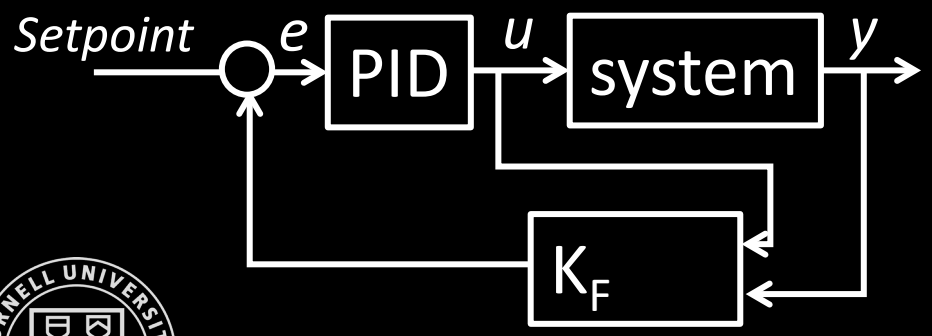
- Controllability
- $\mathbb{C} = [B \quad AB \quad A^2B \quad \dots \quad A^{n-1}B]$
- >>ctrb(A,B)
- Reachability



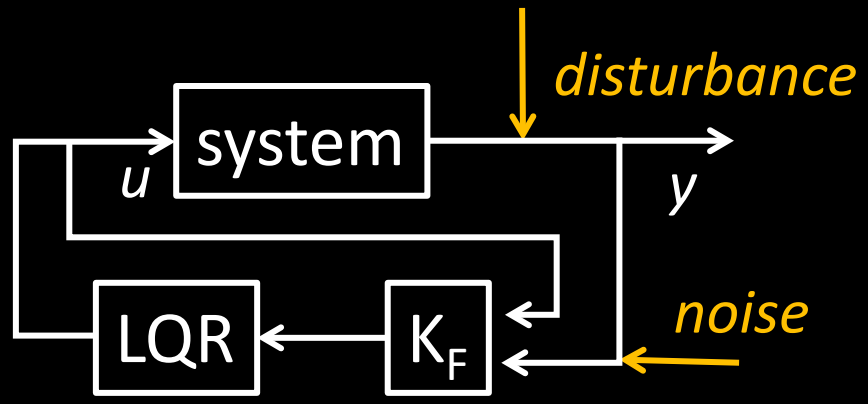
Prof. Kirstin Hagelskjær Petersen
kirstin@cornell.edu

Kalman Filter

KF with PID:



What you typically apply KF on:



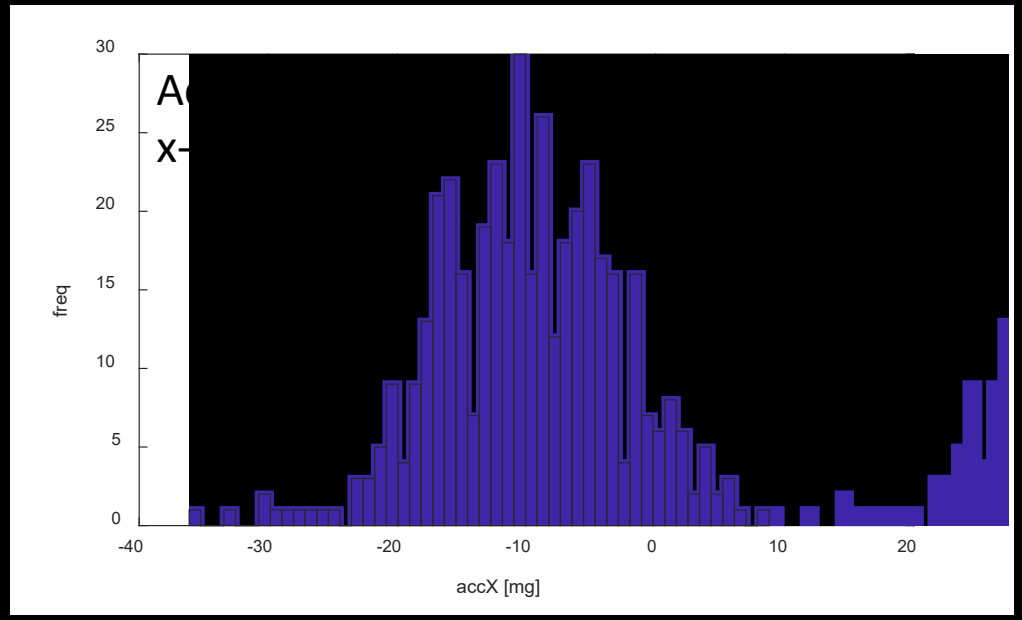
Why KF?

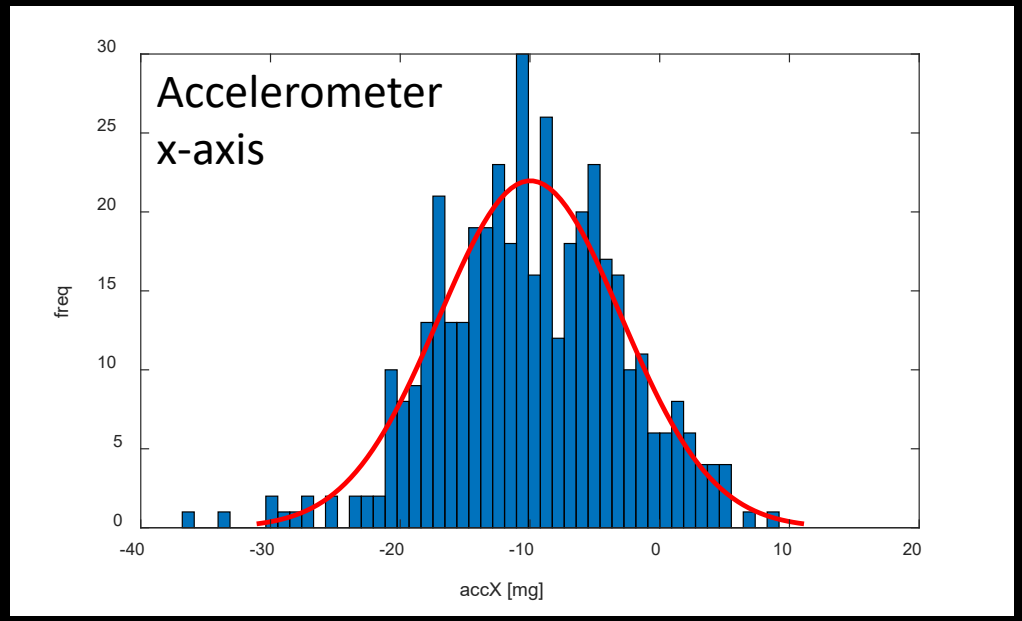
- Not full state feedback
- Bad sensors
- Slow feedback

Probabilistic Robotics

- Measurements are uncertain
- Actions are uncertain
- Models are uncertain
- States are uncertain

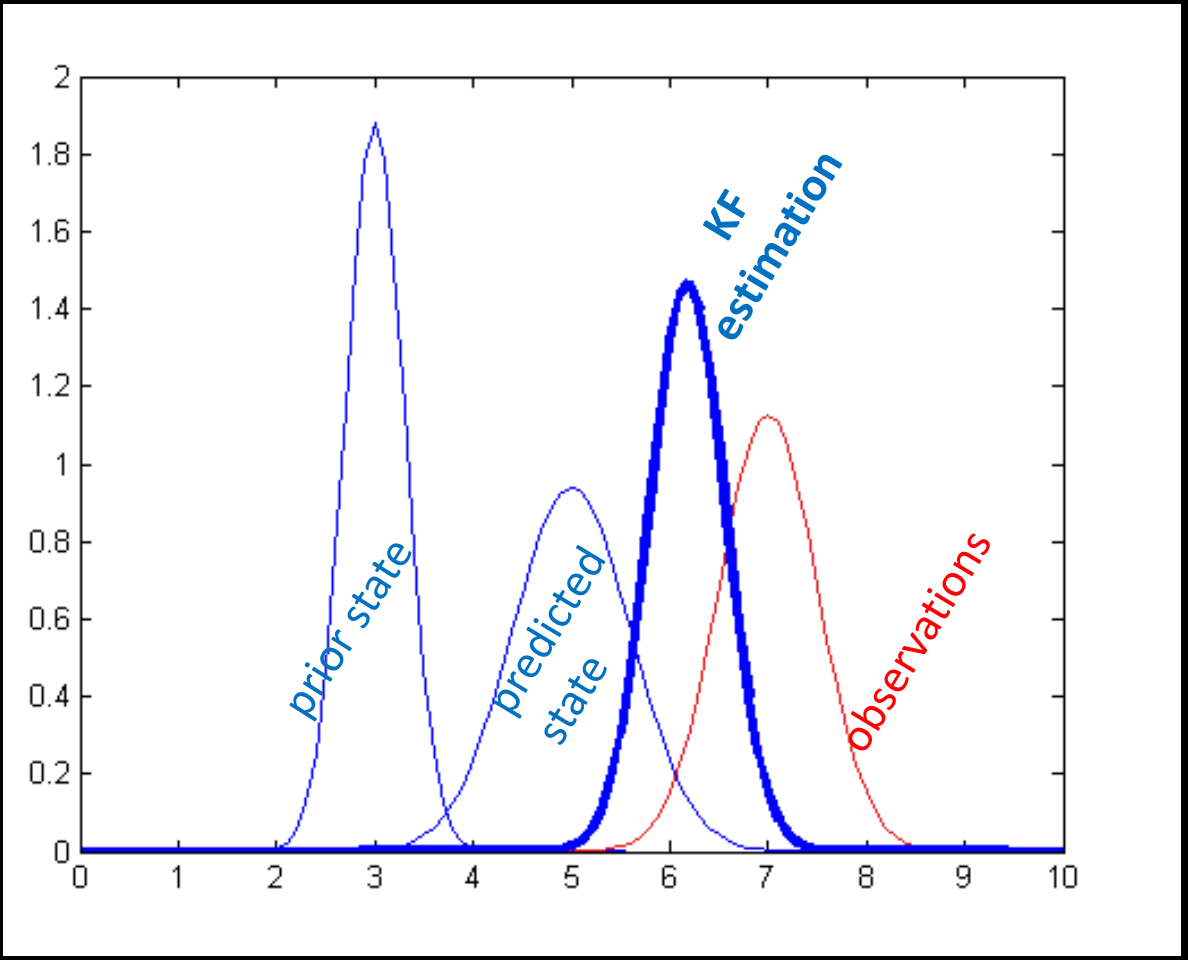
- Noise example: Accelerometer
- Solution?
  - Take more samples
  - Mean: μ = -9.97306mg
  - std dev: σ =7.0318mg
  - Variance: σ^2
- Gaussian distributions
  - $[\mu \mp \sigma]$
  - Symmetric
  - Unimodal
  - Sum to “unity”





Kalman Filter

- Incorporate uncertainty to get better estimates based on inputs and observations



Bayes Filter / Kalman Filter

- Bayes Filter
- Kalman Filter uses the same idea, but uses Gaussian variables for posterior and prior beliefs to speed up computation.

Prior belief input observations

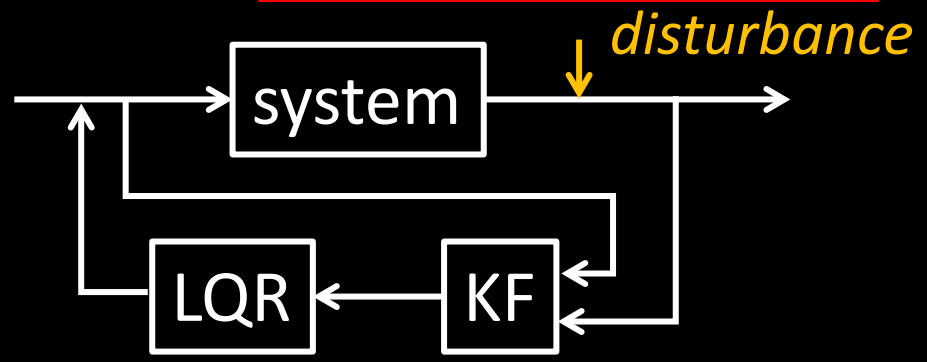
Bayes Filter($bel(x_{t-1})$, u_t, z_t)

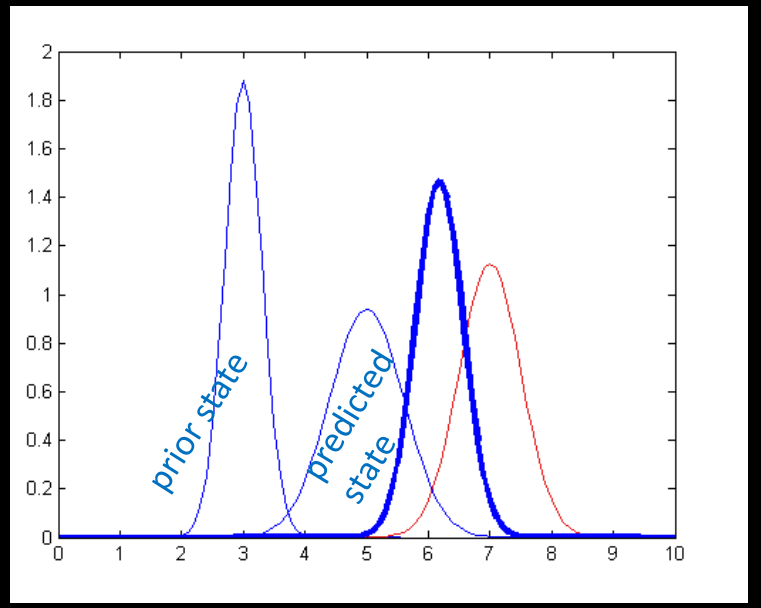
1. for all x(t) do
2. $\overline{bel}(x(t)) = \Sigma\, x(t-1)\, p(x(t) \mid u(t), x(t-1))\, bel(x(t-1))$ Prediction step
3. $bel(x(t)) = \alpha\, p(z(t) \mid x(t))\, \overline{bel}(x(t))$ Update step
4. end for
5. return $bel(x_t)$

Kalman Filter

State estimate: $\mu(t)$
State uncertainty: $\Sigma(t)$
Process noise: Σ_u

- Assume that posterior and prior belief are Gaussian variables
  - Prediction step
    - $x(t) = A\,x(t-1) + Bu(t) + n$, where…
      - $\mu_p(t) = A\,\mu(t-1) + B\,u(t)$
      - $\Sigma_p(t) = A\,\Sigma(t-1)\,A^T + \Sigma_u$

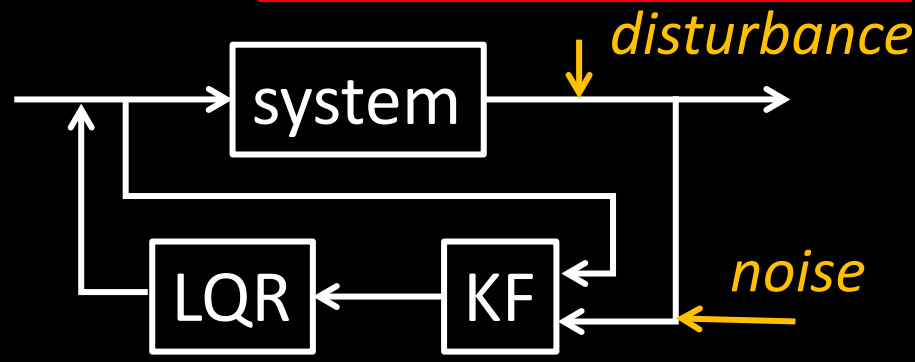


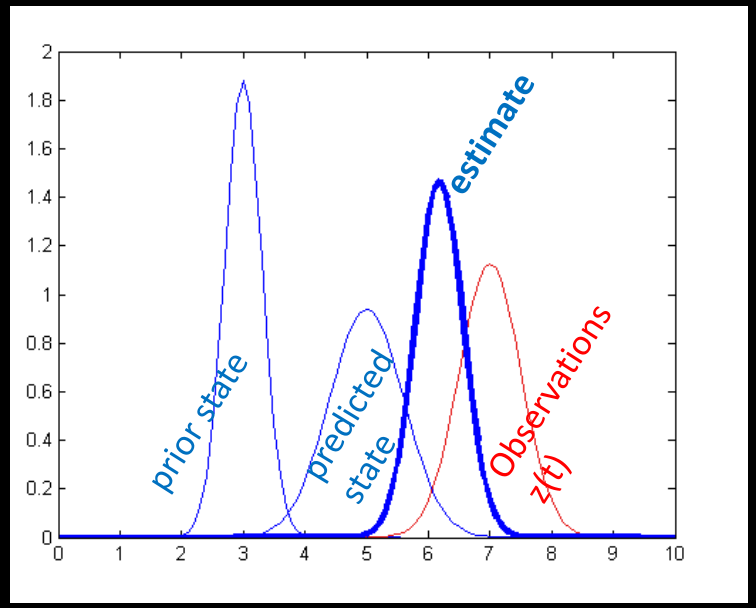


Bayes Filter – Kalman Filter

State estimate: $\mu(t)$
State uncertainty: $\Sigma(t)$
Process noise: Σ_u
Kalman filter gain: K_{KF}
Measurement noise: Σ_z

- Assume that posterior and prior belief are Gaussian variables
  - Prediction step
    - $x(t) = A\,x(t-1) + Bu(t) + n$, where…
      - $\mu_p(t) = A\,\mu(t-1) + B\,u(t)$
      - $\Sigma_p(t) = A\,\Sigma(t-1)\,A^T + \Sigma_u$
  - Update step
    - $K_{KF} = \Sigma_p(t)\,C^T\,(C\,\Sigma_p(t)\,C^T + \Sigma_z)^{-1}$
    - $\mu(t) = \mu_p(t) + K_{KF}\,(z(t) - C\,\mu_p(t))$
    - $\Sigma(t) = (I - K_{KF}\,C)\,\Sigma_p(t)$



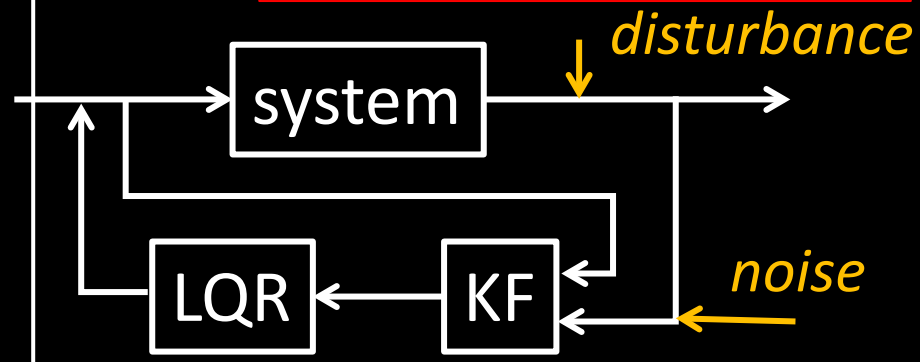


Kalman Filter Implementation

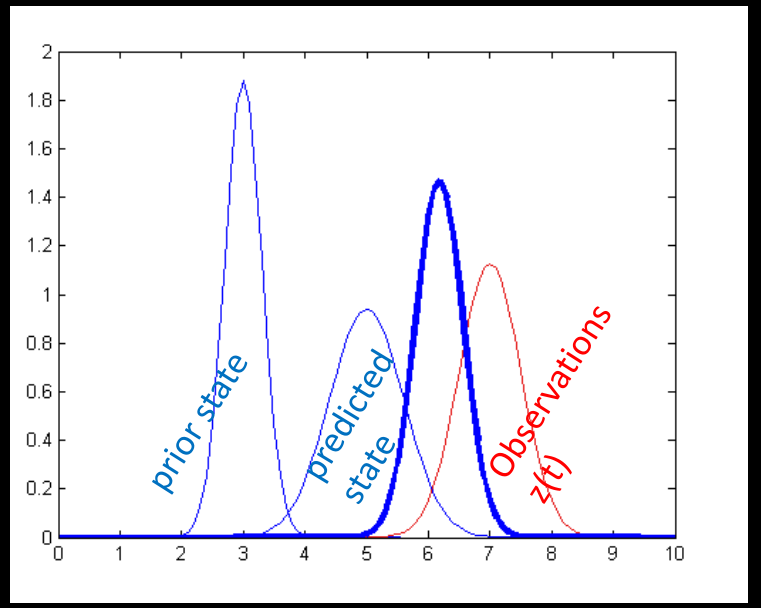
Kalman Filter ($\mu(t-1)$, $\Sigma(t-1)$, $u(t)$, $z(t)$)

1. $\mu_p(t) = A\,\mu(t-1) + B\,u(t)$
2. $\Sigma_p(t) = A\,\Sigma(t-1)\,A^T + \Sigma_u$ } prediction
3. $K_{KF} = \Sigma_p(t)\,C^T\,(C\,\Sigma_p(t)\,C^T + \Sigma_z)^{-1}$
4. $\mu(t) = \mu_p(t) + K_{KF}\,(z(t) - C\,\mu_p(t))$
5. $\Sigma(t) = (I - K_{KF}\,C)\,\Sigma_p(t)$ } update
6. Return $\mu(t)$ and $\Sigma(t)$

State estimate: $\mu(t)$
State uncertainty: $\Sigma(t)$
Process noise: Σ_u
Kalman filter gain: K_{KF}
Measurement noise: Σ_z



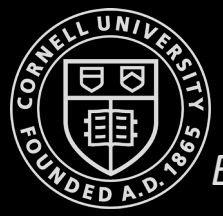
$$\Sigma_u = \begin{bmatrix} {\sigma_1}^2 & 0 & 0 \\ 0 & {\sigma_2}^2 & 0 \\ 0 & 0 & {\sigma_3}^2 \end{bmatrix}, \Sigma_z = \begin{bmatrix} {\sigma_4}^2 & 0 \\ 0 & {\sigma_5}^2 \end{bmatrix}$$



Prof. Kirstin Hagelskjær Petersen
kirstin@cornell.edu

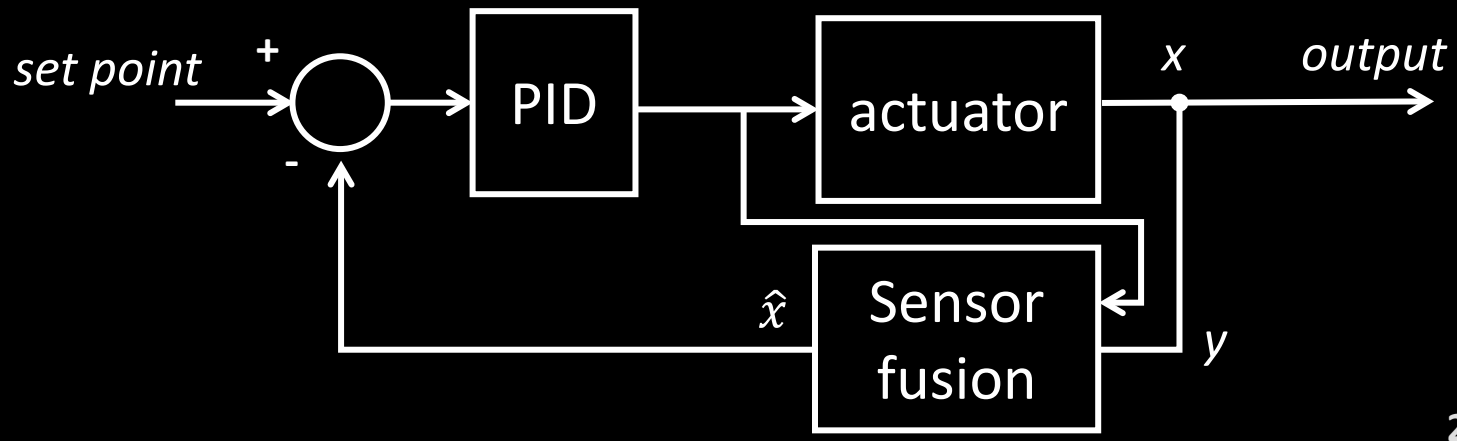
Lab Prep

- Lab 6: PID control
- Lab 7: Sensor Fusion
- Lab 8: Stunt



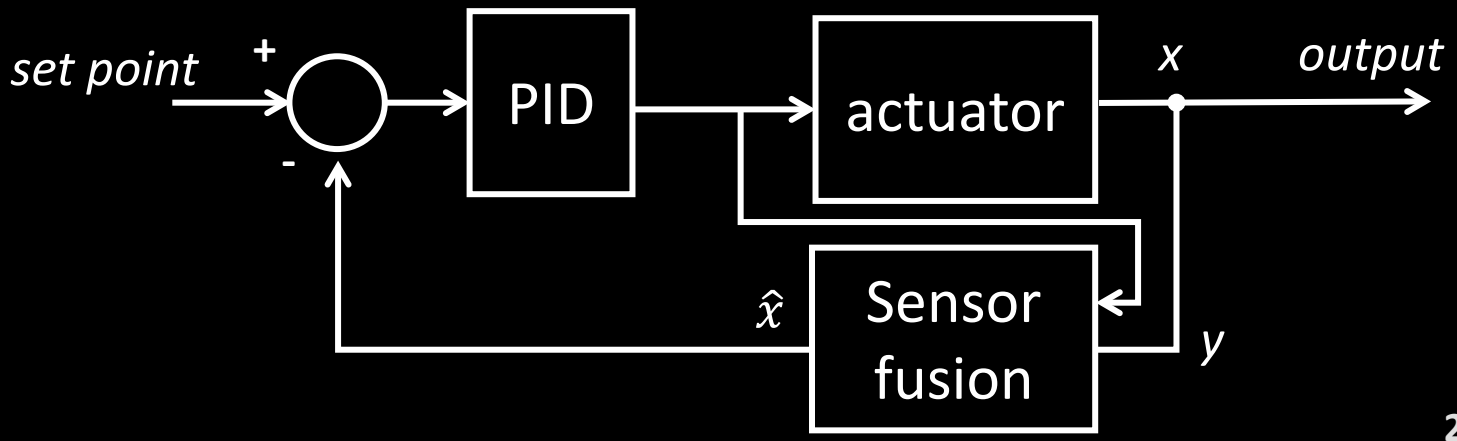
Lab 6-8: PID control – Sensor Fusion - Stunt

- Task A: Don’t Hit the Wall!



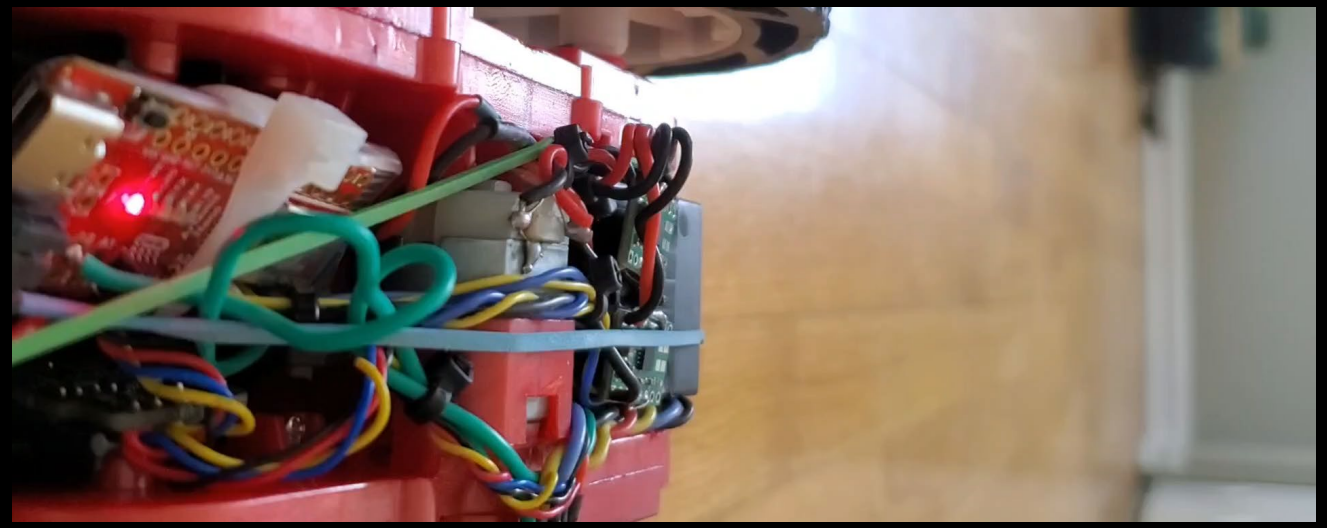
Lab 6-8: PID control – Sensor Fusion - Stunt

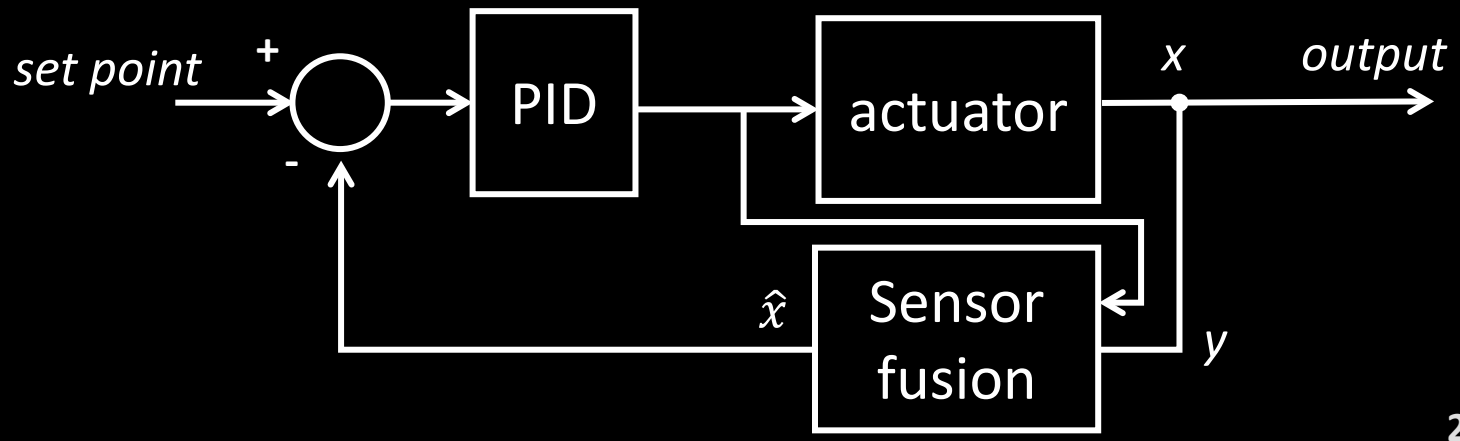
- Task A: Don’t Hit the Wall!
- Task B: Drift much?



Lab 6-8: PID control – Sensor Fusion - Stunt

- Task A: Don’t Hit the Wall!
- Task B: Drift much?
- Task C: Thread the Needle!



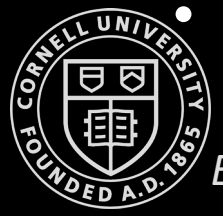


Lab 6-8: PID control – Sensor Fusion - Stunt

- Task A: Don’t Hit the Wall!
- Task B: Drift much?
- Task C: Thread the Needle!

Procedure

- Lab 6: Get basic PID to work
- Do the pre-lab: you need good debugging scripts
- Start simple and work your way up, then hack away…
  - Start slow (sampling rates, control frequency)
  - Avoid blocking statements
- Wind-up, derivative LPF, derivative kick
- Motor scaling function
  - Range of analogWrite: [0;255]
  - Directionality
  - Deadband

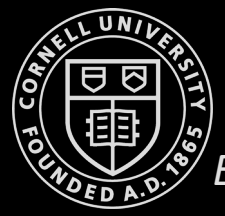


Lab 6-8: PID control – Sensor Fusion - Stunt

- Task A: Don’t Hit the Wall!
- Task B: Drift much?
- Task C: Thread the Needle!

Procedure

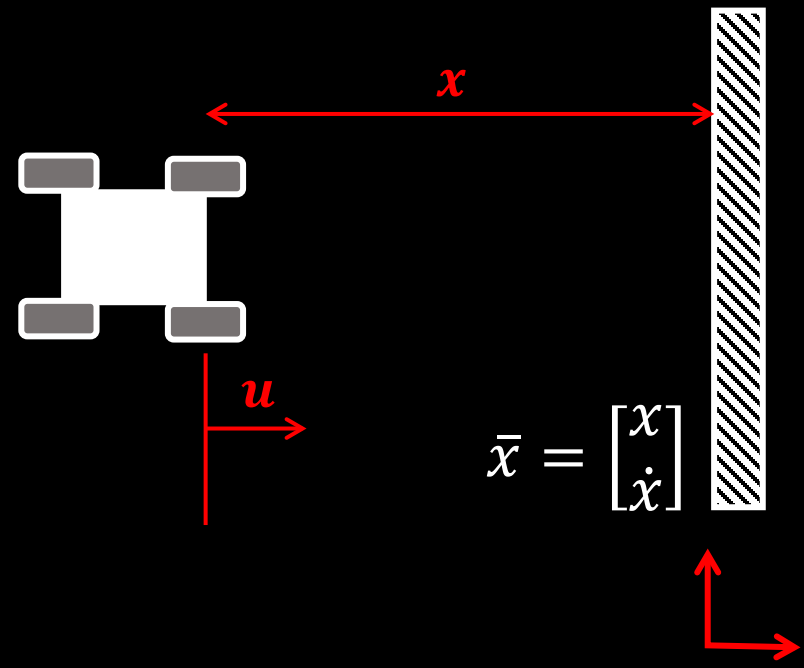
- Lab 6: Get basic PID to work
- Lab 7: Sensor Fusion
  - Approximate the state space equations
    - Step response
  - Implement Kalman Filter
    - Determine process and measurement noise
    - Try it locally
    - Try it onboard
  - Lab 8: Use KF and PID control to execute fast stunts



Lab 7, Task A: State Space Equations

$$F = ma = m\ddot{x}$$
$$F = u - d\dot{x}$$
$$u - d\dot{x} = m\ddot{x}$$
$$\ddot{x} = \frac{u}{m} - \frac{d}{m}\dot{x}$$

What is d and m?



State space equation

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{d}{m} \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u$$

$$C = \begin{bmatrix} -1 & 0 \end{bmatrix}$$

Lab 7, Task A: State Space Equations

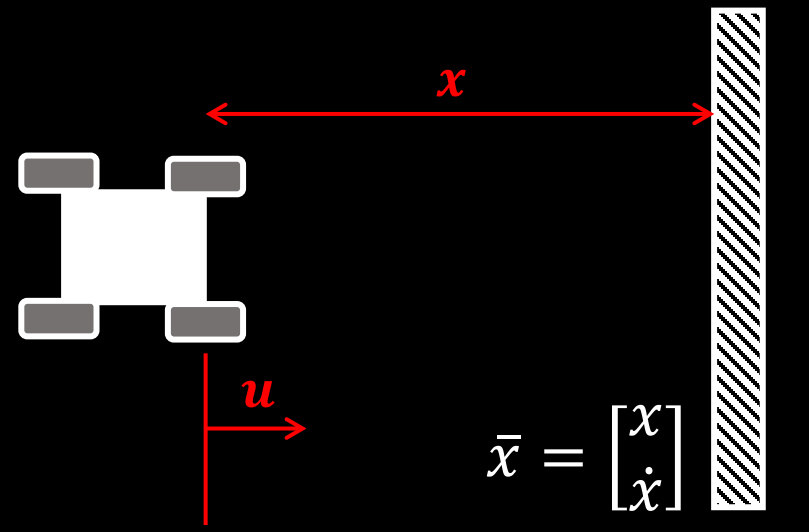
$$F = ma = m\ddot{x}$$
$$F = u - d\dot{x}$$
$$u - d\dot{x} = m\ddot{x}$$
$$\ddot{x} = \frac{u}{m} - \frac{d}{m}\dot{x}$$

1st order system:

$$\frac{dy(t)}{dt} + \frac{1}{\tau}y(t) = x(t)$$

Step response solution:

$$y(t) = 1 - e^{-\frac{t}{\tau}}$$



$$\bar{x} = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$$

What is d and m?

- At steady state (cst speed), we can find *d*
  - $0 = \frac{u}{m} - \frac{d}{m}\dot{x}$
  - $0 = \frac{u}{m} - \frac{d}{m}\dot{x} \quad \leftrightarrow \quad d = \frac{u}{\dot{x}}$
- Find *m*
  - $\dot{v} = \frac{u}{m} - \frac{d}{m}v$
- Step response solution for a first order system
  - $m = \frac{-dt_{0.9}}{\ln(0.1)}$

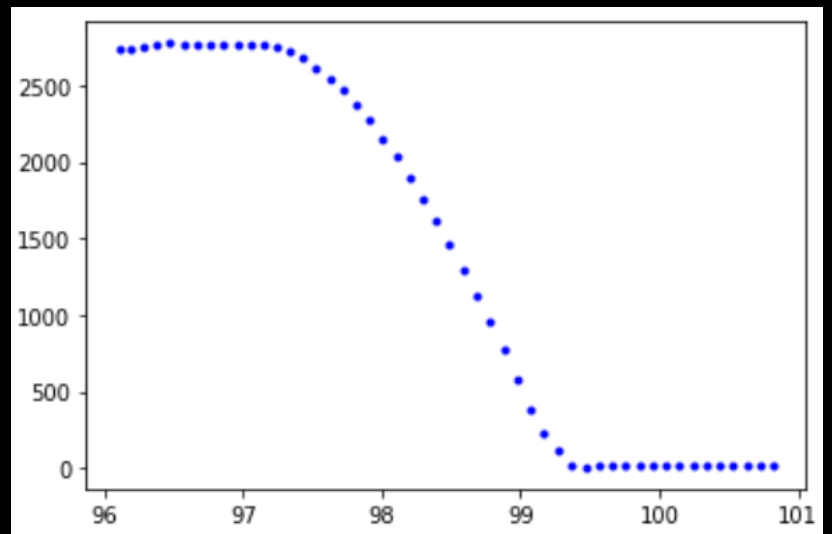
State space equation

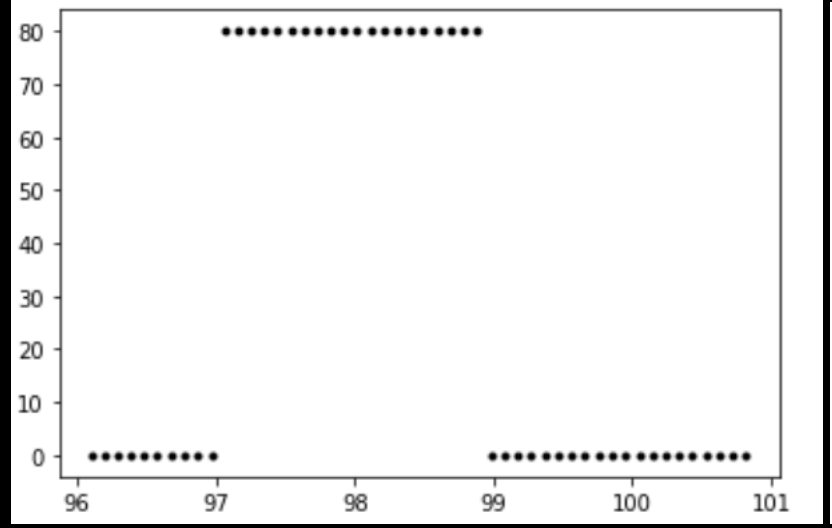
$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{d}{m} \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u$$

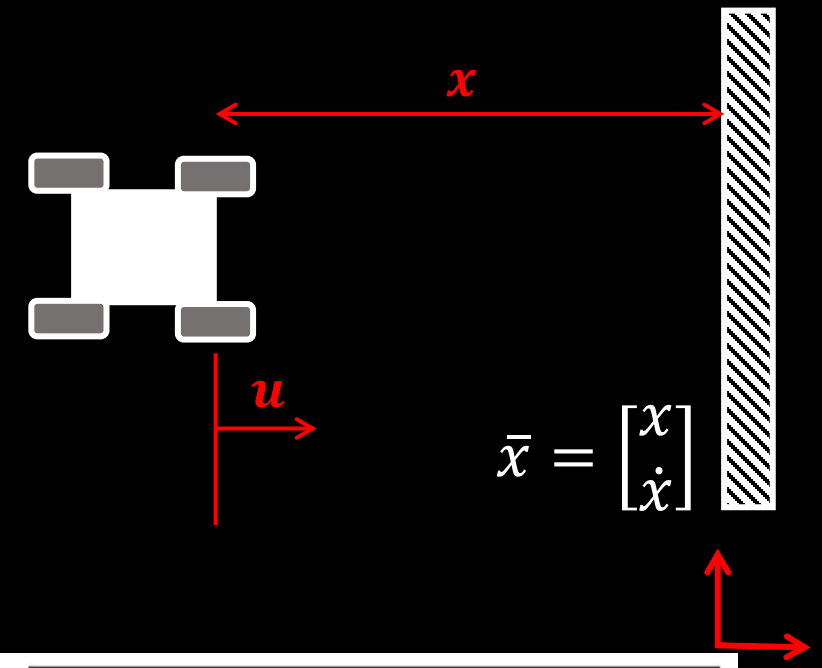
$$C = [-1 \quad 0]$$

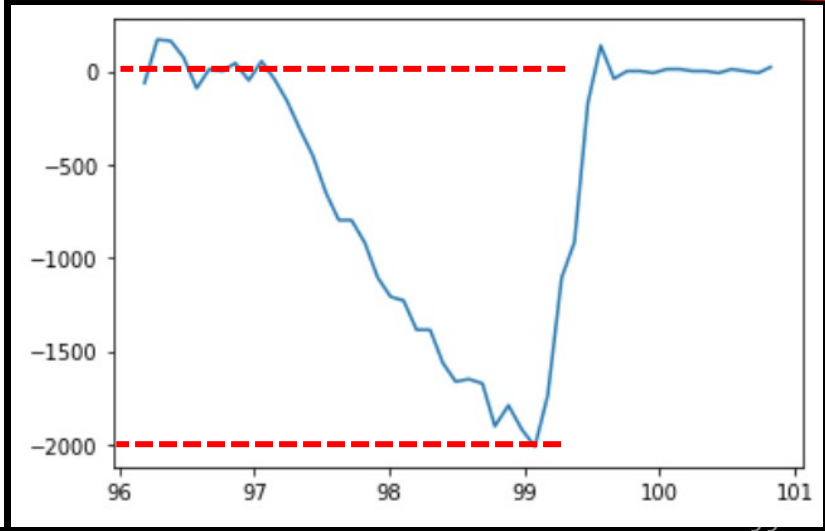
Lab 7, Task A: State Space Equations

- Step response
  - Drag (d)
    - Steady state speed
  - Momentum (m)
    - Rise time



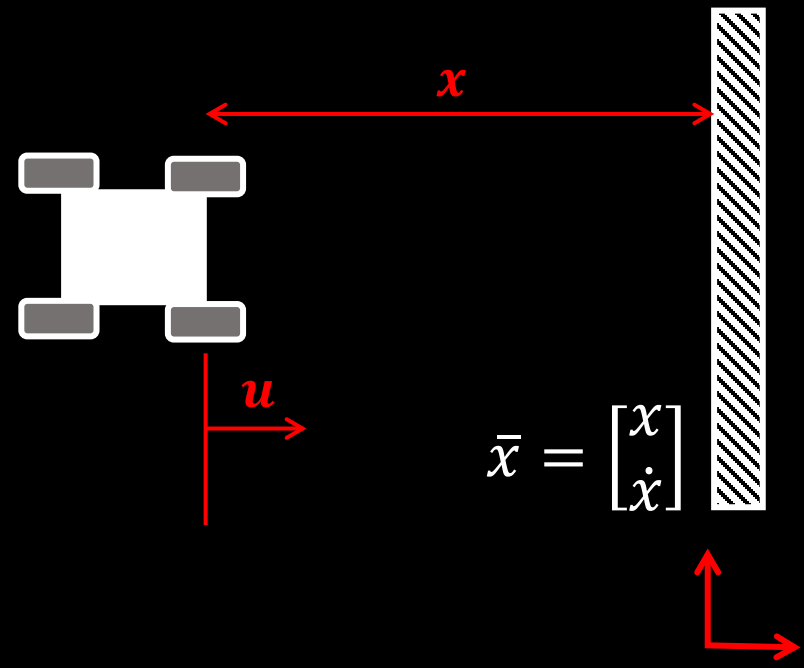






Lab 7, Task A: State Space Equations

$$F = ma = m\ddot{x}$$
$$F = u - d\dot{x}$$
$$u - d\dot{x} = m\ddot{x}$$
$$\ddot{x} = \frac{u}{m} - \frac{d}{m}\dot{x}$$



What is d and m?

- At steady state (cst speed), we can find *d*
  - $d = \frac{u}{\dot{x}} \approx \frac{1}{2000mm/s}$
- Find *m*
  - $m = \frac{-dt_{0.9}}{\ln(0.1)} \approx \frac{-0.0005 \cdot 1.9s}{\ln(0.1)} = 4.1258e\text{-}04$

State space equation

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{d}{m} \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u$$

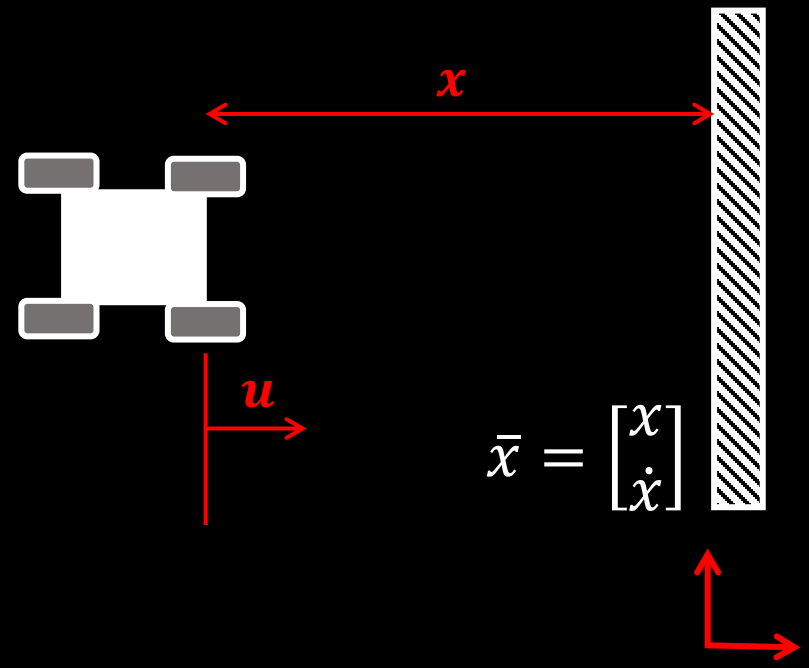
$$C = [-1 \quad 0]$$

Lab 7, Task A: State Space Equations

- Implement the Kalman Filter
  - Process noise (dependent on sampling rate)

$$\Sigma_u = \begin{bmatrix} {\sigma_1}^2 & 0 \\ 0 & {\sigma_2}^2 \end{bmatrix}$$

  - Pos: $\sigma_1^2 = 5mm$, 50 times per second:
    - $\sigma = \sqrt{5^2 \cdot 50} = 35mm$
  - Speed: $\sigma_2^2 = 10mm/s$, 50 times per second:
    - $\sigma = \sqrt{10^2 \cdot 50} = 71mm/s$
  - Measurement noise
    - $\Sigma_z = [{\sigma_3}^2]$
    - $\sigma_3^2 = 20mm^2$



State space equation

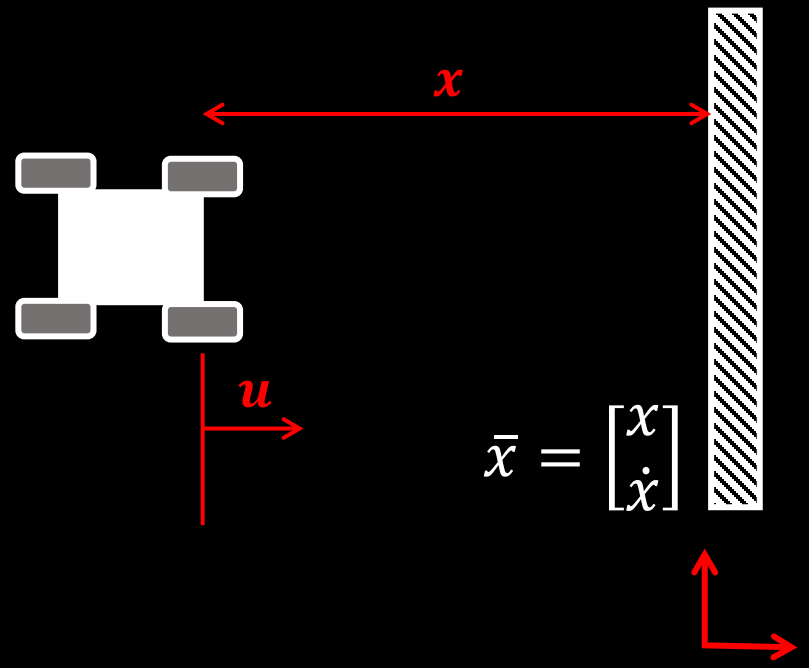
$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{d}{m} \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u$$

$$C = [-1 \quad 0]$$

Lab 7, Task A: State Space Equations

- We have A, B, Σ_u, Σ_z
- Discretize the A and B matrices!
  - x(n+1) = x(n) + dx
  - dx = dt (Ax + Bu)
  - x(n+1) = x(n) + dt (Ax(n) + Bu)
  - x(n+1) = (I + dt*A) x(n) + dt*B u

A_d B_d

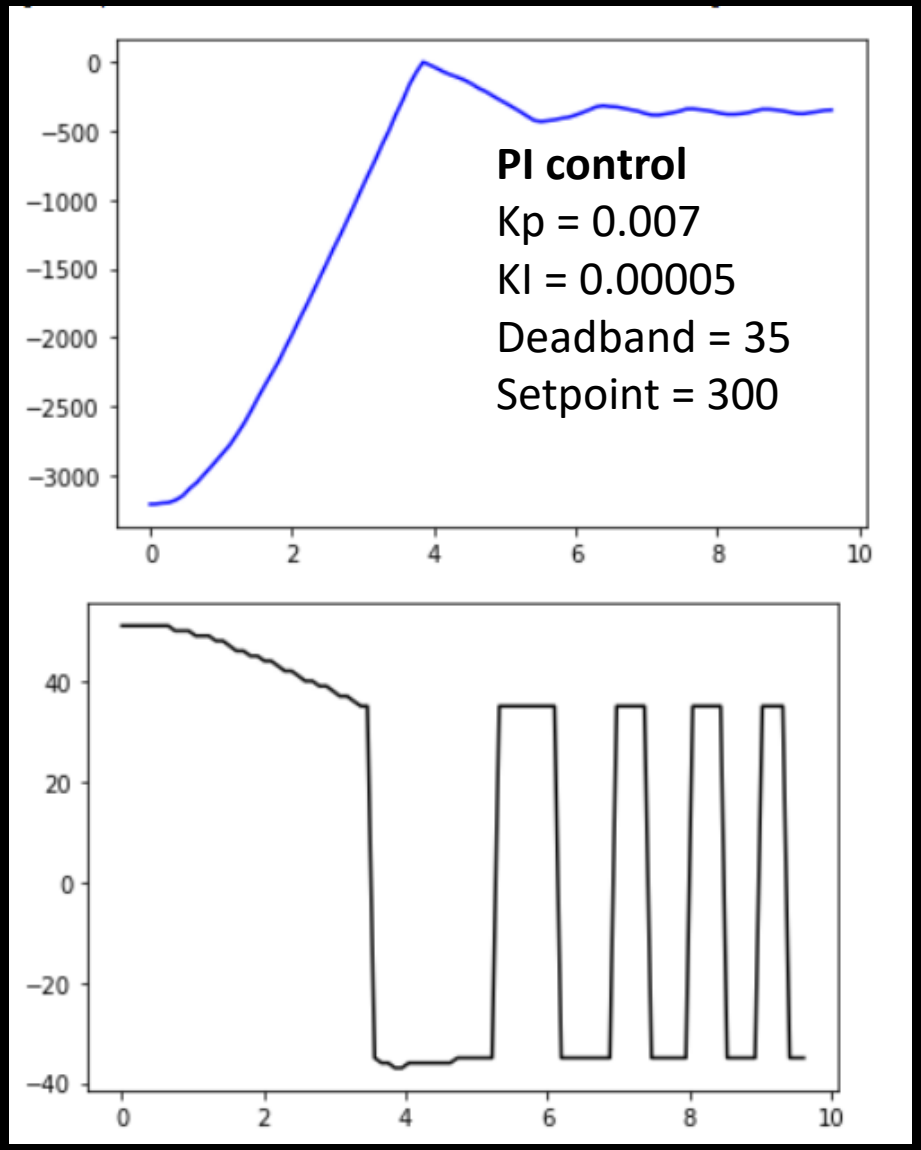


State space equation

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{d}{m} \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u$$

$$C = \begin{bmatrix} -1 & 0 \end{bmatrix}$$

Lab 7, Task A: Kalman Filter



With Kalman filter