Practical Course: Vision-based Navigation Winter Term 2016/2017

Lecture 3: State Estimation and Control

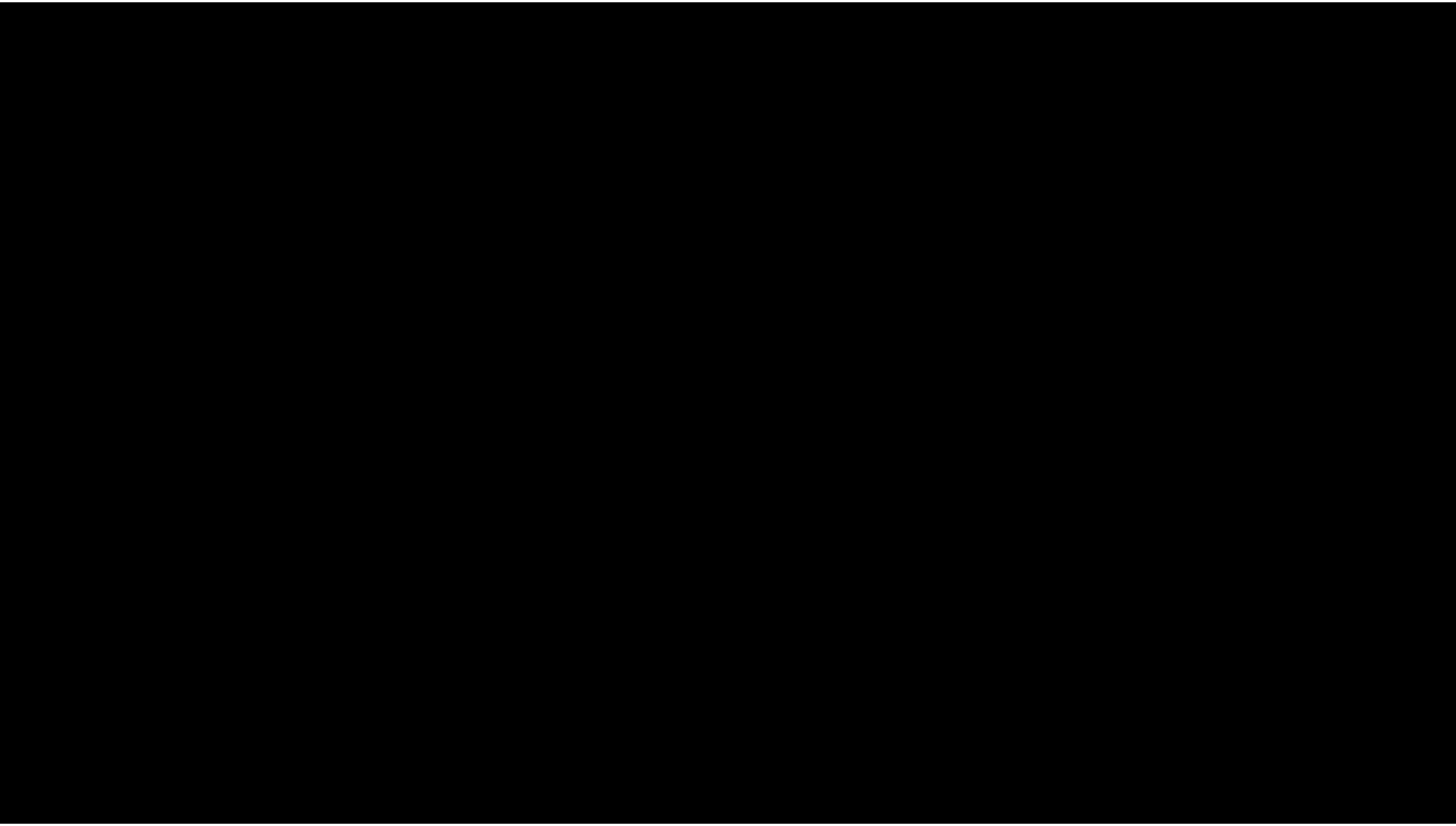
Vladyslav Usenko, Benjamin Grzimek,
Lukas von Stumberg, Prof. Dr. Daniel Cremers

What we will cover today

- Introduction to vision-based state estimation and control

- State estimation
  - Bayes Filter
  - Extended Kalman Filter
  - Unscented Kalman Filter

- Control
  - PID Control
  - Cascaded Control



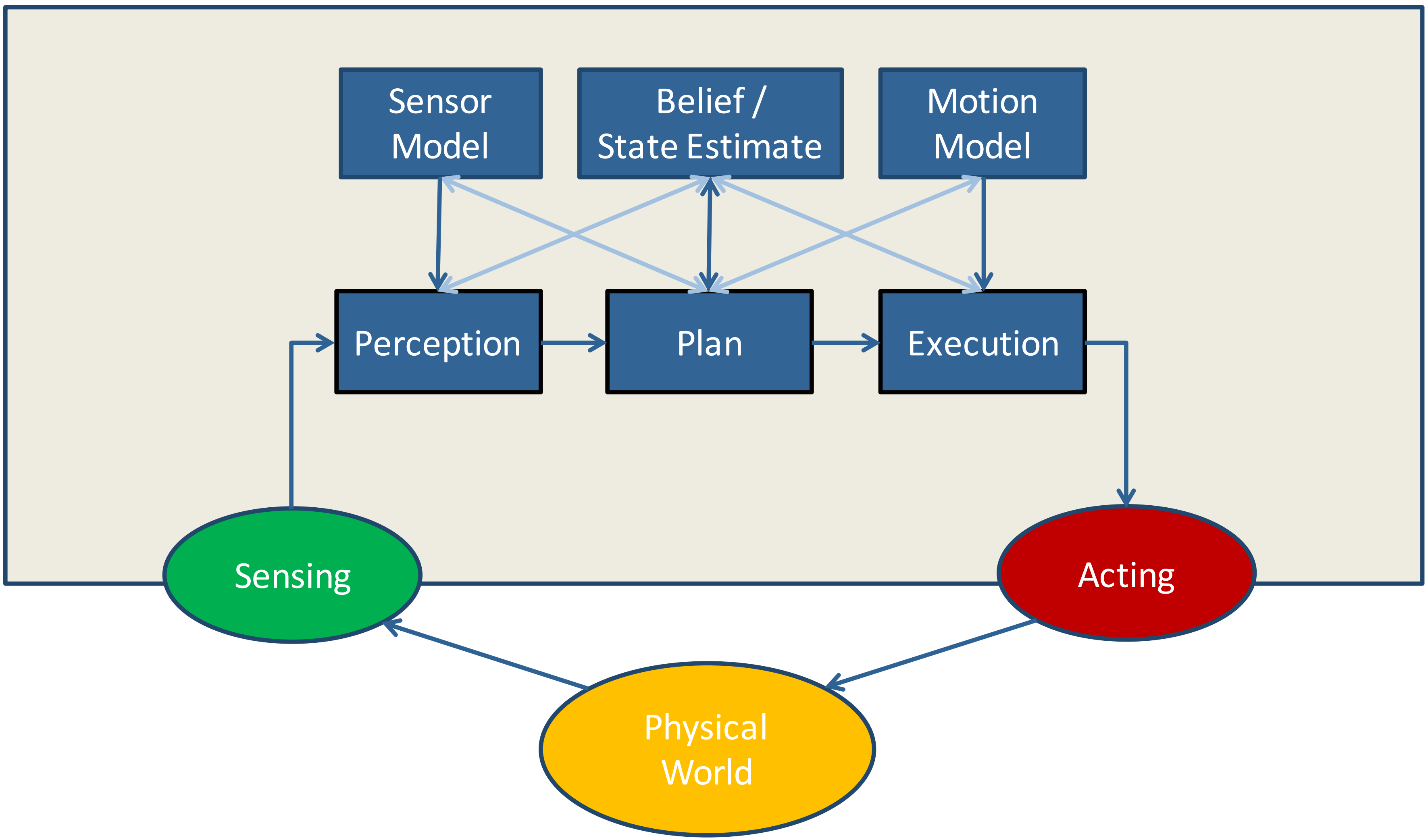
What we will cover today

- Introduction to vision-based state estimation and control

- **State estimation**
  - **Bayes Filter**
  - **Extended Kalman Filter**
  - **Unscented Kalman Filter**

- Feedback Control
  - PID Control
  - Cascaded Control

Models, State Estimation and Control



The State Estimation Problem

We want to estimate the world state x_t from

1. Sensor measurements $z_{1:t}$ and
2. Controls (or odometry readings) $u_{1:t}$

Probabilistic filtering: $p(x_t \mid u_{1:t}, z_{1:t})$

- How do we perform inference for the state?
- How do we model the relationship between these random variables?

Probabilistic Measurement Model

- Measurements depend on the actual state, but robot sensors only provide noisy versions
- Quantify probability distribution on measurements (given state)

$$p(z_t \mid x_t)$$

- Typical model: non-linear function of state and additive noise

$$z_t = h(x_t) + \delta_t \quad \text{e.g.} \qquad \delta_t \sim \mathcal{N}(0, R)$$

z_t: sensor reading
h: measurement function
x_t: world state

Probabilistic State-Transition Model

- Robot executes a control not accurately, i.e. the control outcome can only be predicted up to some uncertainty
- Quantify probability on control outcome (given prev. state)

$$p(x_t \mid x_{t-1}, u_t)$$

- Typical model: non-linear function of control and prev. state with additive noise

state-transition function ↓ executed control ↓

$$x_t = g(x_{t-1}, u_t) + \epsilon_t \quad \text{e.g.} \quad \epsilon_t \sim \mathcal{N}(0, Q)$$

current state ↑ previous state ↑

Bayes Filter

- Given:
  - Stream of measurements and controls: $z_{1:t}$ $u_{1:t}$
  - Measurement model $p(z_t \mid x_t)$
  - State-transition model $p(x_t \mid x_{t-1}, u_t)$
  - Prior probability of the system state $p(x_0)$

- Wanted:
  - Estimate of the state x_t of the dynamic system
  - Posterior of the state is also called **belief**

$$\text{Bel}\,(x_t) = p(x_t \mid u_{1:t}, z_{1:t})$$

Markov Assumption

- Measurements depend only on current state

$$p(z_t \mid x_{0:t}, z_{1:t-1}, u_{1:t}) = p(z_t \mid x_t)$$

- Current state depends only on prev. state and current control

$$p(x_t \mid x_{0:t}, z_{1:t}, u_{1:t}) = p(x_t \mid x_{t-1}, u_t)$$

- Underlying assumptions
  - Static world
  - Independent noise
  - Perfect model, no approximation errors

Bayes Filter

For each time step, do

1. Apply motion model

$$\overline{\text{Bel}}\,(x_t) = \int_{x_{t-1}} p(x_t \mid x_{t-1}, u_t)\,\text{Bel}\,(x_{t-1})\,dx_{t-1}$$

2. Apply sensor model

$$\text{Bel}\,(x_t) = \eta\, p(z_t \mid x_t)\,\overline{\text{Bel}}\,(x_t)$$

Kalman Filter

- Bayes filter with
  - continuous states
  - Gaussian state variable and model noise
  - Linear measurement and state-transition functions
  - Extension to non-linear models (Extended Kalman Filter EKF)
- Developed in the late 1950's
- Kalman filter is very efficient (only requires a few matrix operations per time step)
- Applications range from economics, weather forecasting, satellite navigation to robotics and many more
- Most relevant Bayes filter variant in practice

Normal Distribution

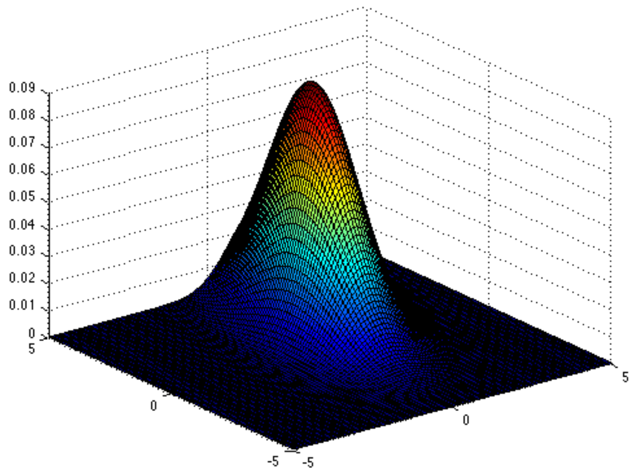
- Multivariate normal distribution

$$X \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$$

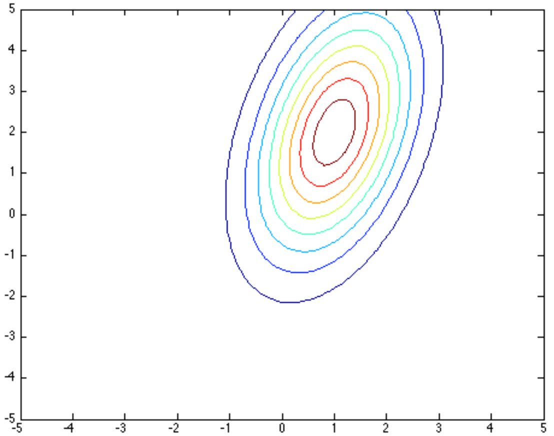
$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \Sigma) = \frac{1}{(2\pi)^{d/2}\ |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$$

- Example: 2-dimensional normal distribution

pdf



iso lines



Properties of Normal Distributions

- Linear transformation → remains Gaussian

$$X \sim \mathcal{N}(\mu, \Sigma), Y \sim AX + B$$
$$\Rightarrow Y \sim \mathcal{N}(A\mu + B, A\Sigma A^\top)$$

- Intersection of two Gaussians → remains Gaussian

$$X_1 \sim \mathcal{N}(\mu_1, \Sigma_1), X_2 \sim \mathcal{N}(\mu_2, \Sigma_2)$$
$$\Rightarrow p(X_1, X_2) = \mathcal{N}\left(\frac{\Sigma_2}{\Sigma_1 + \Sigma_2}\mu_1 + \frac{\Sigma_1}{\Sigma_1 + \Sigma_2}\mu_2, \frac{1}{\Sigma_1^{-1} + \Sigma_2^{-1}}\right)$$

Kalman Filter

Estimates the state x_t of a discrete-time controlled process that is governed by the linear stochastic difference equation

$$x_t = Ax_{t-1} + Bu_t + \epsilon_t$$

and (linear) measurements of the state

$$z_t = Cx_t + \delta_t$$

with $\delta_t \sim \mathcal{N}(0, R)$ and $\epsilon_t \sim \mathcal{N}(0, Q)$

Initial belief is Gaussian $\mathrm{Bel}(x_0) = \mathcal{N}(x_0; \mu_0, \Sigma_0)$

From Bayes Filter to Kalman Filter

For each time step, do

1. Apply state-transition model

$$\overline{\text{Bel}}(x_t) = \int \underbrace{p(x_t \mid x_{t-1}, u_t)}_{\mathcal{N}(x_t; Ax_{t-1}+Bu_t, Q)} \underbrace{\text{Bel}(x_{t-1})}_{\mathcal{N}(x_{t-1}; \mu_{t-1}, \Sigma_{t-1})} \, \mathrm{d}x_{t-1}$$

$$= \mathcal{N}(x_t; A\mu_{t-1} + Bu_t, A\Sigma A^\top + Q)$$

$$= \mathcal{N}(x_t; \bar{\mu}_t, \bar{\Sigma}_t)$$

From Bayes Filter to Kalman Filter

For each time step, do

2. Apply measurement model

$$\text{Bel}(x_t) = \eta \underbrace{p(z_t \mid x_t)}_{\mathcal{N}(z_t;Cx_t,R)} \underbrace{\overline{\text{Bel}}(x_t)}_{\mathcal{N}(x_t;\bar{\mu}_t,\bar{\Sigma}_t)}$$

$$= \mathcal{N}(x_t; \bar{\mu}_t + K_t(z_t - C\bar{\mu}), (I - K_t C)\bar{\Sigma})$$

$$= \mathcal{N}(x_t; \mu_t, \Sigma_t)$$

with $\quad K_t = \bar{\Sigma}_t C^\top (C\bar{\Sigma}_t C^\top + R)^{-1}$

Kalman Filter

For the interested readers:
See Probabilistic Robotics for full derivation (Chapter 3)

For each time step, do

1. Apply state-transition model

$$\bar{\mu}_t = A\mu_{t-1} + Bu_t$$
$$\bar{\Sigma}_t = A\Sigma A^\top + Q$$

2. Apply measurement model

$$\mu_t = \bar{\mu}_t + K_t(z_t - C\bar{\mu}_t)$$
$$\Sigma_t = (I - K_t C)\bar{\Sigma}_t$$

with

$$K_t = \bar{\Sigma}_t C^\top (C\bar{\Sigma}_t C^\top + R)^{-1}$$

Kalman Filter

- Highly efficient: Polynomial in the measurement dimensionality *k* and state dimensionality *n*:

$$O(k^{2.376} + n^2)$$

- **Optimal for linear Gaussian systems**!

- Most robotics systems are **nonlinear**!
(i.e. nonlinear measurement and state-transition model)

Taylor Expansion

- Solution: Linearize both functions
- State-transition function

$$\begin{aligned} g(x_{t-1}, u_t) &\approx g(\mu_{t-1}, u_t) + \frac{\partial g(\mu_{t-1}, u_t)}{\partial x_{t-1}}(x_{t-1} - \mu_{t-1}) \\ &= g(\mu_{t-1}, u_t) + G_t(x_{t-1} - \mu_{t-1}) \end{aligned}$$

- Measurement function

$$\begin{aligned} h(x_t) &\approx h(\bar{\mu}_t) + \frac{\partial h(\bar{\mu}_t)}{\partial x_t}(x_t - \mu_t) \\ &= h(\bar{\mu}_t) + H_t(x_t - \mu_t) \end{aligned}$$

Extended Kalman Filter

For the interested readers:
See Probabilistic Robotics for full derivation (Chapter 3)

For each time step, do

1. Apply state-transition model

$$\bar{\mu}_t = g(\mu_{t-1}, u_t)$$
$$\bar{\Sigma}_t = G_t \Sigma G_t^\top + Q \quad \text{with} \quad G_t = \frac{\partial g(\mu_{t-1}, u_t)}{\partial x_{t-1}}$$

2. Apply measurement model

$$\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$$
$$\Sigma_t = (I - K_t H_t)\bar{\Sigma}_t$$

with $K_t = \bar{\Sigma}_t H_t^\top (H_t \bar{\Sigma}_t H_t^\top + R)^{-1}$ and $H_t = \frac{\partial h(\bar{\mu}_t)}{\partial x_t}$

Example

- 2D case
- State $\mathbf{x} = \begin{pmatrix} x & y & \psi \end{pmatrix}^\top$
- Odometry $\mathbf{u} = \begin{pmatrix} \dot{x} & \dot{y} & \dot{\psi} \end{pmatrix}^\top$
- Measurements $\mathbf{z} = \left(z_x \, z_y \, z_\theta \right)^T$ (relative to robot pose) of visual marker at position $\mathbf{l} = \left(l_x \, l_y \right)^T$
- Fixed time intervals Δt

Example

- State-transition function

$$g(\mathbf{x}, \mathbf{u}) = \begin{pmatrix} x + (\cos(\psi)\dot{x} - \sin(\psi)\dot{y})\Delta t \\ y + (\sin(\psi)\dot{x} + \cos(\psi)\dot{y})\Delta t \\ \psi + \dot{\psi}\Delta t \end{pmatrix}$$

- Derivative of state-transition function

$$G = \frac{\partial g(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}} = \begin{pmatrix} 1 & 0 & (-\sin(\psi)\dot{x} - \cos(\psi)\dot{y})\Delta t \\ 0 & 1 & (\cos(\psi)\dot{x} + \sin(\dot{\psi})\dot{y})\Delta t \\ 0 & 0 & 1 \end{pmatrix}$$

Example

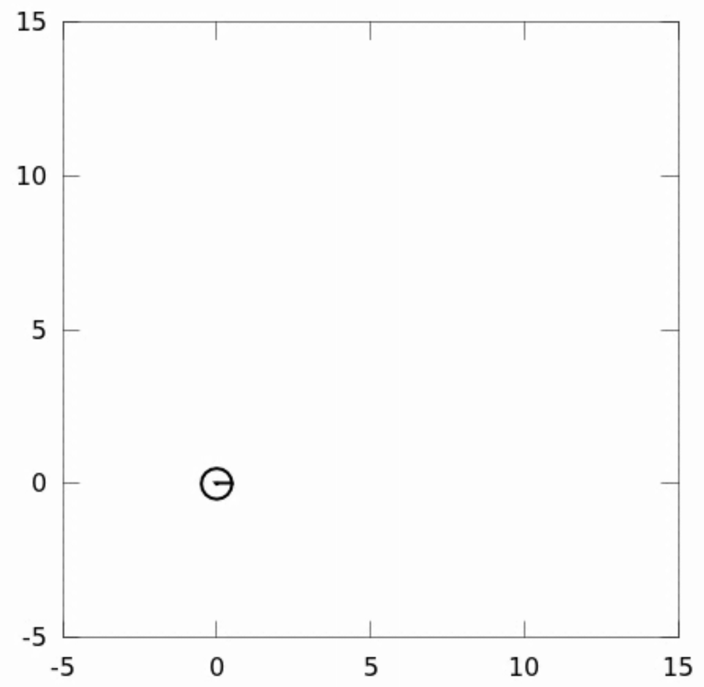
- Measurement function

$$h(\mathbf{x}) = \begin{pmatrix} \mathbf{R}(\psi)^T & 0 \\ \mathbf{0} & 1 \end{pmatrix} \begin{pmatrix} l_x - x \\ l_y - y \\ \arctan\left(\frac{l_y - y}{l_x - x}\right) - \psi \end{pmatrix}$$

$$H = \frac{\partial h(\mathbf{x})}{\partial \mathbf{x}} = \nabla_{\mathbf{x}} \begin{pmatrix} \mathbf{R}(\psi)^T & 0 \\ \mathbf{0} & 1 \end{pmatrix} \begin{pmatrix} l_x - x \\ l_y - y \\ \arctan\left(\frac{l_y - y}{l_x - x}\right) - \psi \end{pmatrix}$$
$$+ \begin{pmatrix} \mathbf{R}(\psi)^T & 0 \\ \mathbf{0} & 1 \end{pmatrix} \nabla_{\mathbf{x}} \begin{pmatrix} l_x - x \\ l_y - y \\ \arctan\left(\frac{l_y - y}{l_x - x}\right) - \psi \end{pmatrix}$$

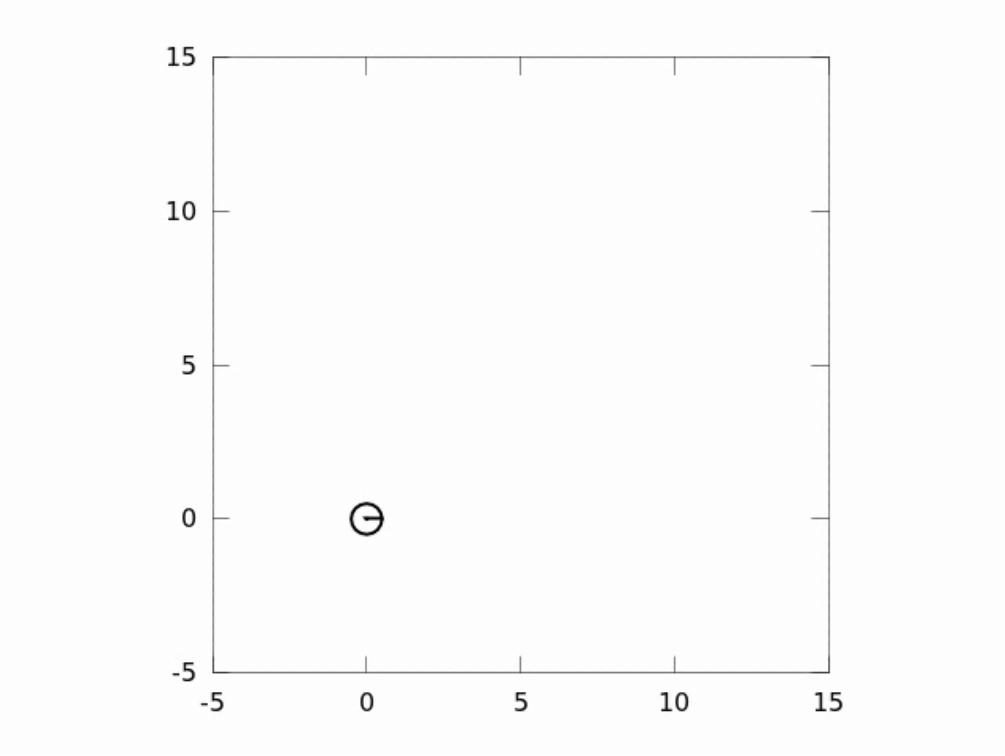
Example

- Dead reckoning (no measurements)
- Large process noise in x+y



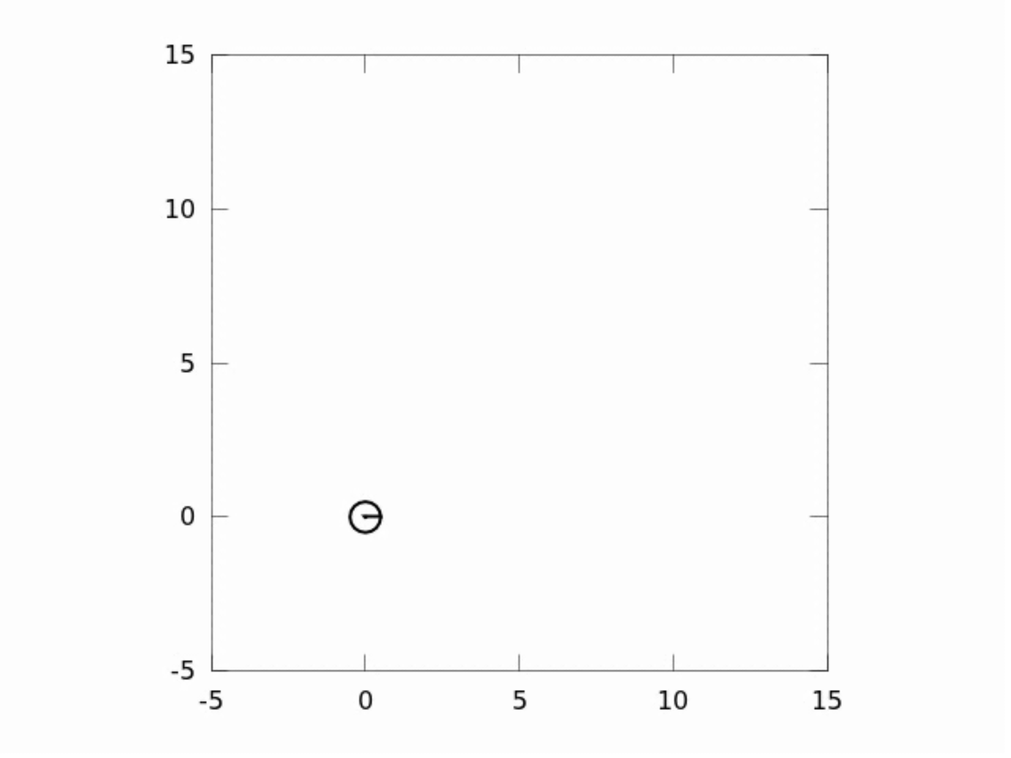
Example

- Dead reckoning (no measurements)
- Large process noise in x+y+yaw



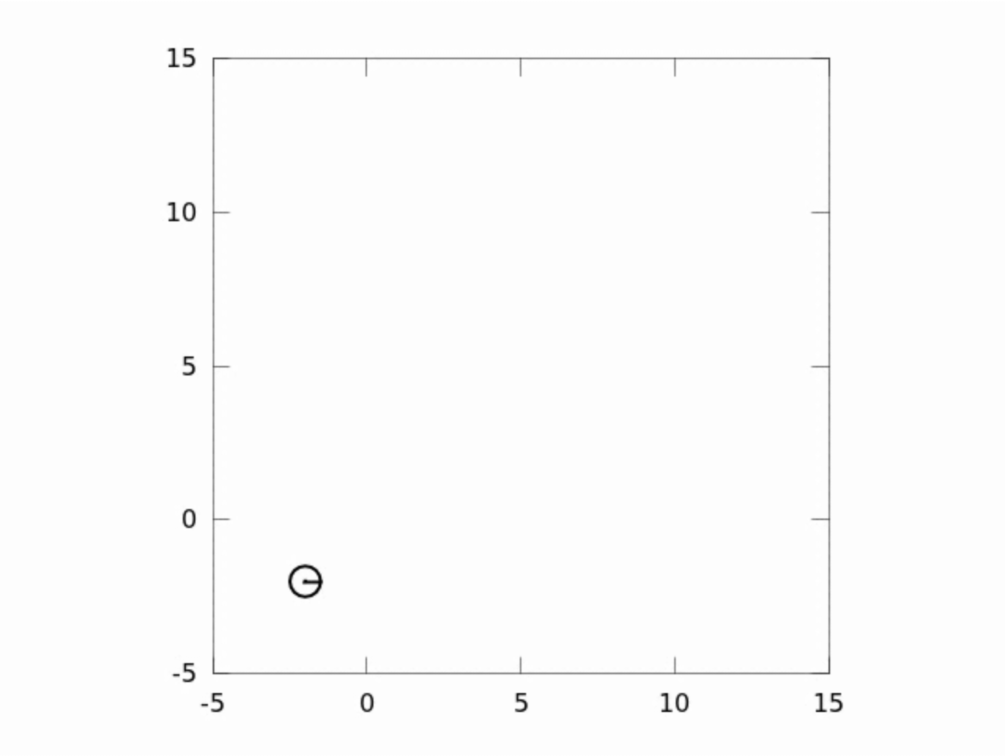
Example

- Now with measurements (limited visibility)
- Assume robot knows correct starting pose



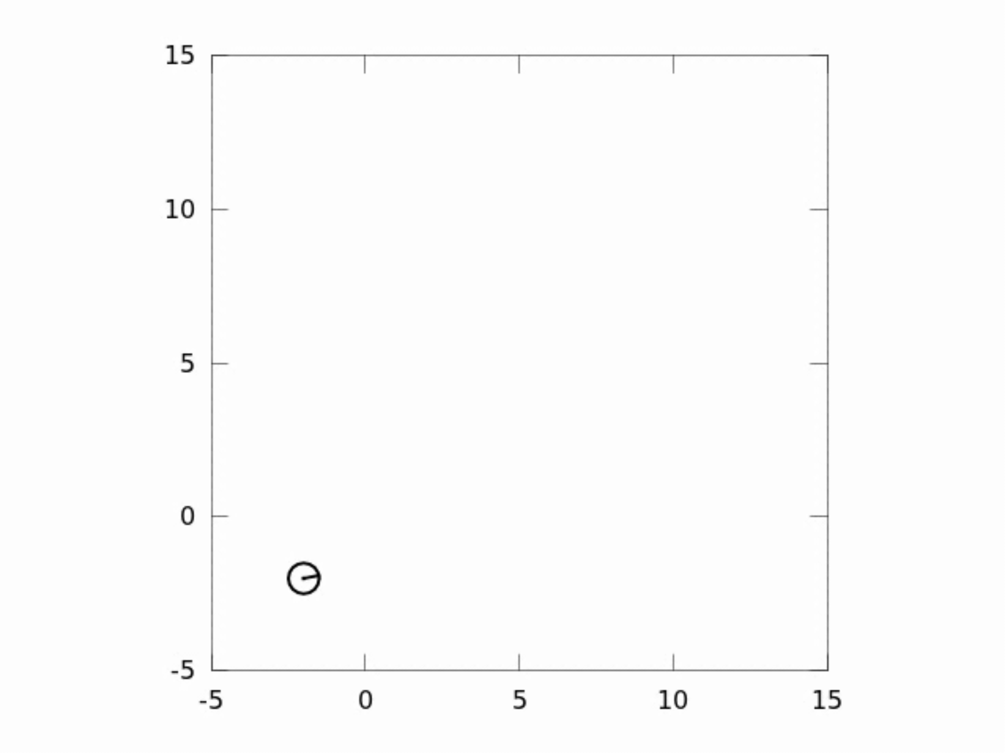
Example

- What if the initial pose (x+y) is wrong?



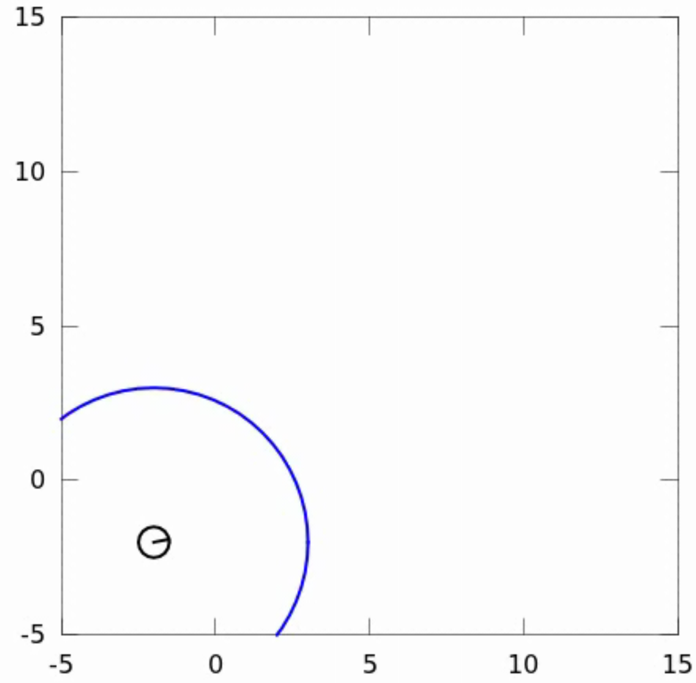
Example

- What if the initial pose (x+y+yaw) is wrong?

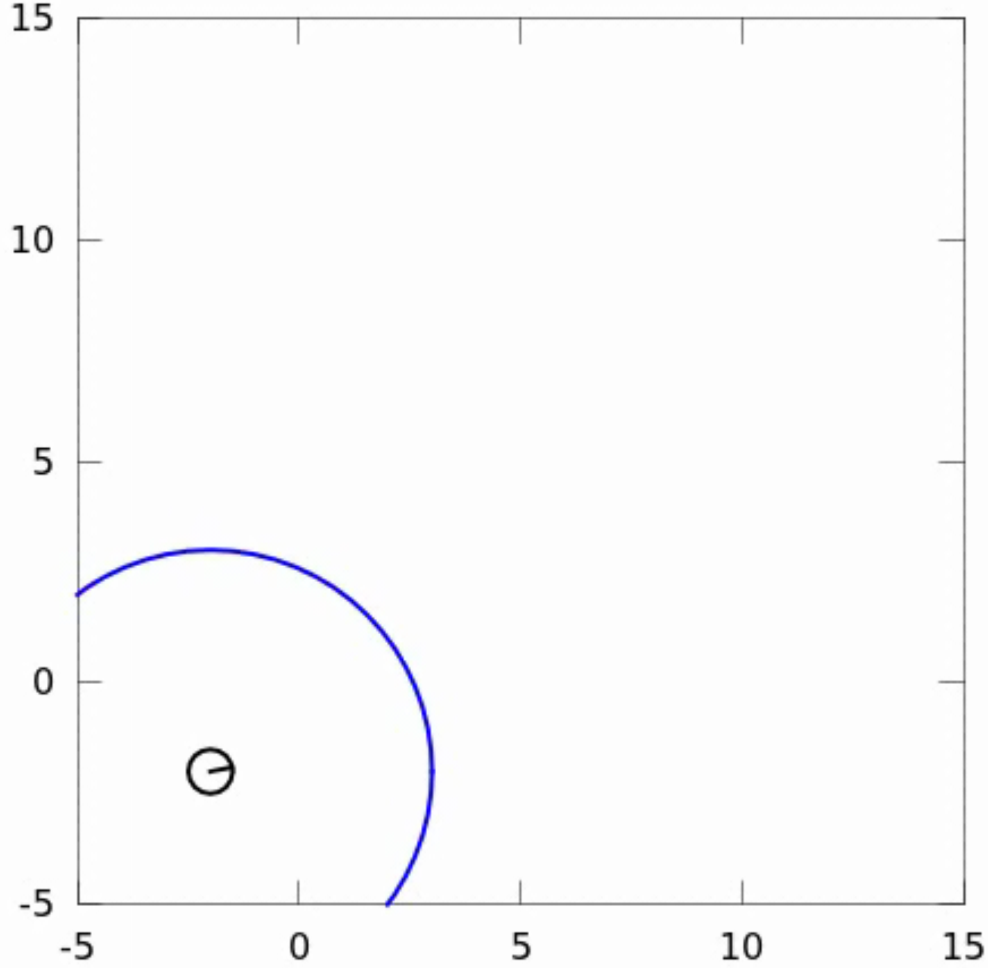


Example

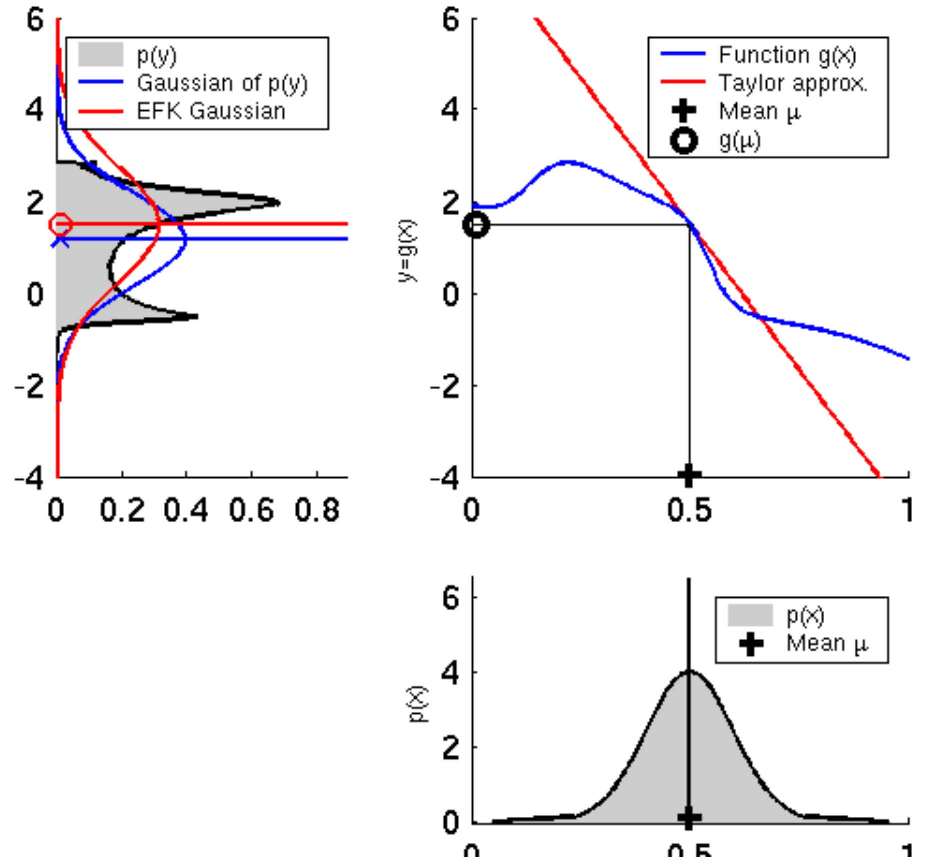
- If we are aware of a bad initial guess, we set the initial covariance to a large value (large uncertainty)



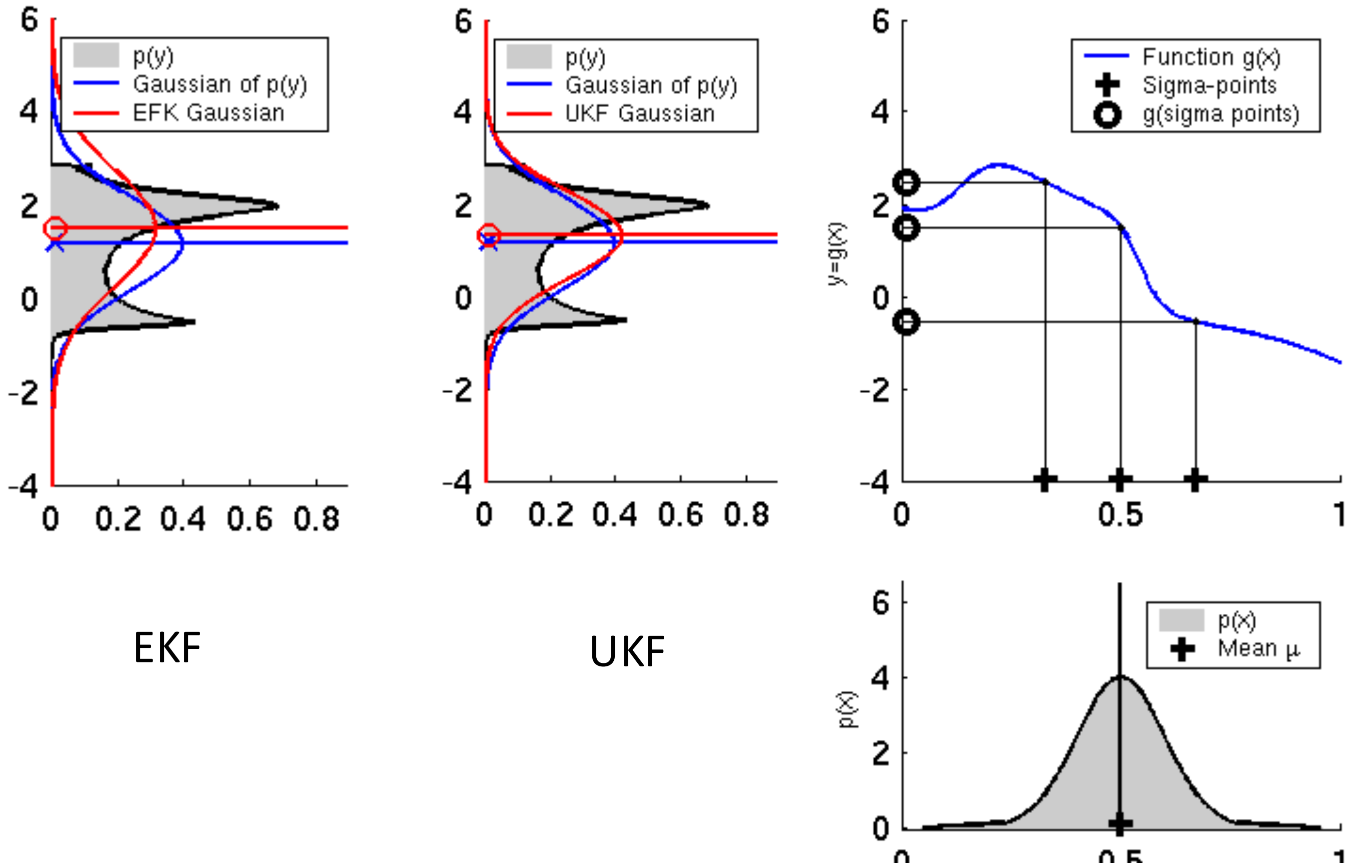
Example



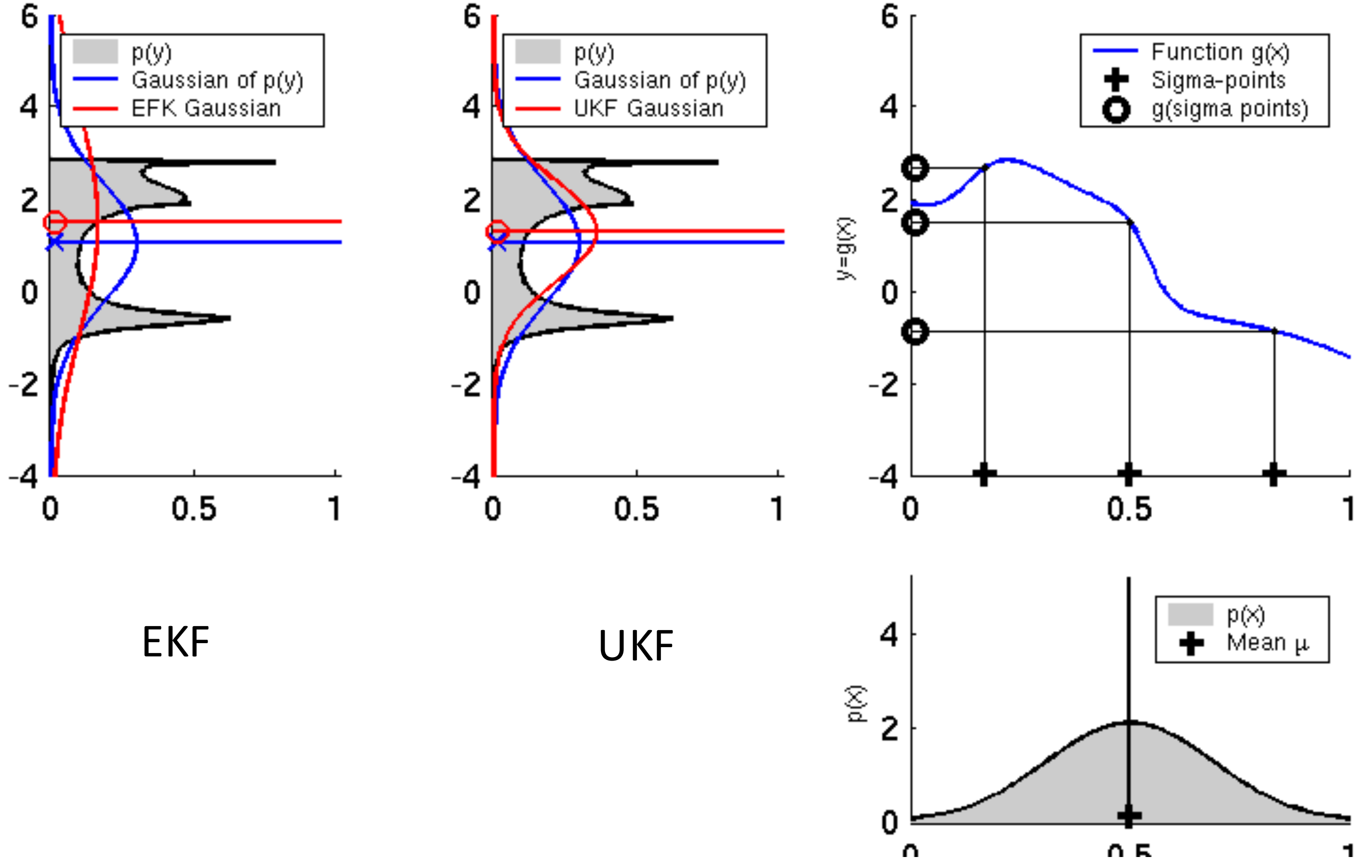
Linearization via EKF



Linearization via Unscented Transform (1)



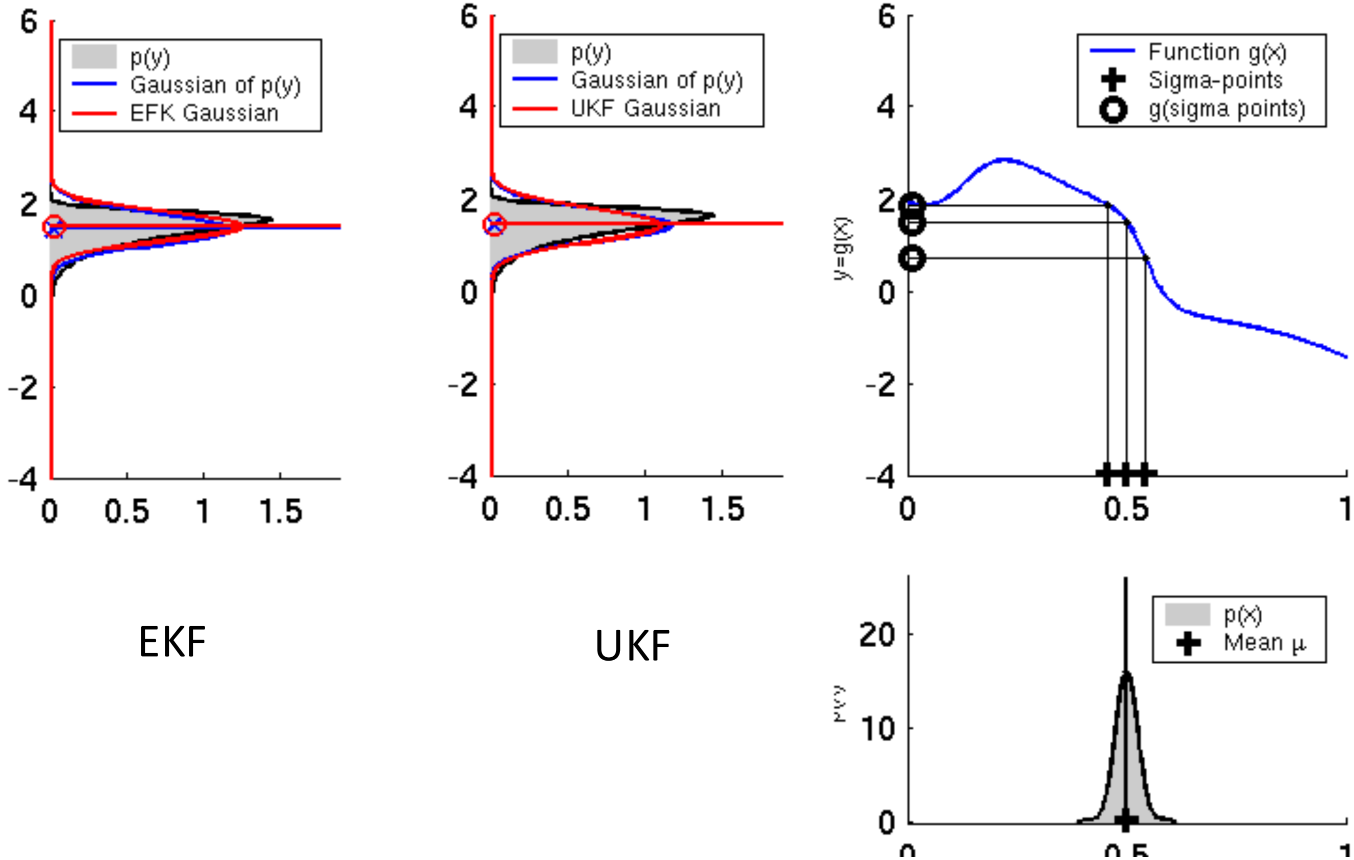
Linearization via Unscented Transform (2)



EKF

UKF

Linearization via Unscented Transform (3)



EKF

UKF

Unscented Transform

Sigma points

$$\chi^0 = \mu$$

matrix square root

$$\chi^i = \mu \pm \left(\sqrt{(n+\lambda)\Sigma}\right)_i$$

Weights

$$w_m^0 = \frac{\lambda}{n+\lambda} \qquad w_c^0 = \frac{\lambda}{n+\lambda} + (1 - \alpha^2 + \beta)$$

$$w_m^i = w_c^i = \frac{1}{2(n+\lambda)} \qquad \text{for } i = 1,\dots,2n$$

Pass sigma points through nonlinear function

$$\psi^i = g(\chi^i)$$

Recover mean and covariance

$$\mu' = \sum_{i=0}^{2n} w_m^i \psi^i$$

$$\Sigma' = \sum_{i=0}^{2n} w_c^i (\psi^i - \mu)(\psi^i - \mu)^T$$

Unscented Transform on SE(3):
C. Hertzberg et al., “Integrating Generic Sensor Fusion Algorithms with Sound State Representations through Encapsulation of Manifolds”, http://arxiv.org/pdf/1107.1119.pdf

Cholesky Decomposition

- Symmetric positive definite matrices (such as covariance) can be factored into

$$\mathbf{\Sigma} = \mathbf{L}\mathbf{L}^T$$

using the Cholesky decomposition.

- The matrix square root is defined as

$$\sqrt{\mathbf{\Sigma}} := \mathbf{L}$$

Unscented Kalman Filter – Prediction Step

1: **Unscented_Kalman_filter**$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$**:**

2: $\mathcal{X}_{t-1} = (\mu_{t-1} \qquad \mu_{t-1} + \gamma\sqrt{\Sigma_{t-1}} \qquad \mu_{t-1} - \gamma\sqrt{\Sigma_{t-1}})$

3: $\bar{\mathcal{X}}_t^* = g(u_t, \mathcal{X}_{t-1})$

4: $\bar{\mu}_t = \sum_{i=0}^{2n} w_m^{[i]} \bar{\mathcal{X}}_t^{*[i]}$

5: $\bar{\Sigma}_t = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{X}}_t^{*[i]} - \bar{\mu}_t)(\bar{\mathcal{X}}_t^{*[i]} - \bar{\mu}_t)^T + R_t$

Unscented Kalman Filter – Correction Step

6: $\bar{\mathcal{X}}_t = (\bar{\mu}_t \qquad \bar{\mu}_t + \gamma\sqrt{\bar{\Sigma}_t} \qquad \bar{\mu}_t - \gamma\sqrt{\bar{\Sigma}_t})$

7: $\bar{\mathcal{Z}}_t = h(\bar{\mathcal{X}}_t)$

8: $\hat{z}_t = \sum_{i=0}^{2n} w_m^{[i]} \bar{\mathcal{Z}}_t^{[i]}$

9: $S_t = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^T + Q_t$

10: $\bar{\Sigma}_t^{x,z} = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{X}}_t^{[i]} - \bar{\mu}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^T$

11: $K_t = \bar{\Sigma}_t^{x,z} \, S_t^{-1}$

12: $\mu_t = \bar{\mu}_t + K_t(z_t - \hat{z}_t)$

13: $\Sigma_t = \bar{\Sigma}_t - K_t \, S_t \, K_t^T$

14: *return* μ_t, Σ_t

Unscented Kalman Filter – Correction Step

6: $\bar{\mathcal{X}}_t = (\bar{\mu}_t \qquad \bar{\mu}_t + \gamma\sqrt{\bar{\Sigma}_t} \qquad \bar{\mu}_t - \gamma\sqrt{\bar{\Sigma}_t})$

7: $\bar{\mathcal{Z}}_t = h(\bar{\mathcal{X}}_t)$

8: $\hat{z}_t = \sum_{i=0}^{2n} w_m^{[i]} \bar{\mathcal{Z}}_t^{[i]}$

9: $S_t = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^T + Q_t$

10: $\bar{\Sigma}_t^{x,z} = \sum_{i=0}^{2n} w_c^{[i]} (\bar{\mathcal{X}}_t^{[i]} - \bar{\mu}_t)(\bar{\mathcal{Z}}_t^{[i]} - \hat{z}_t)^T$

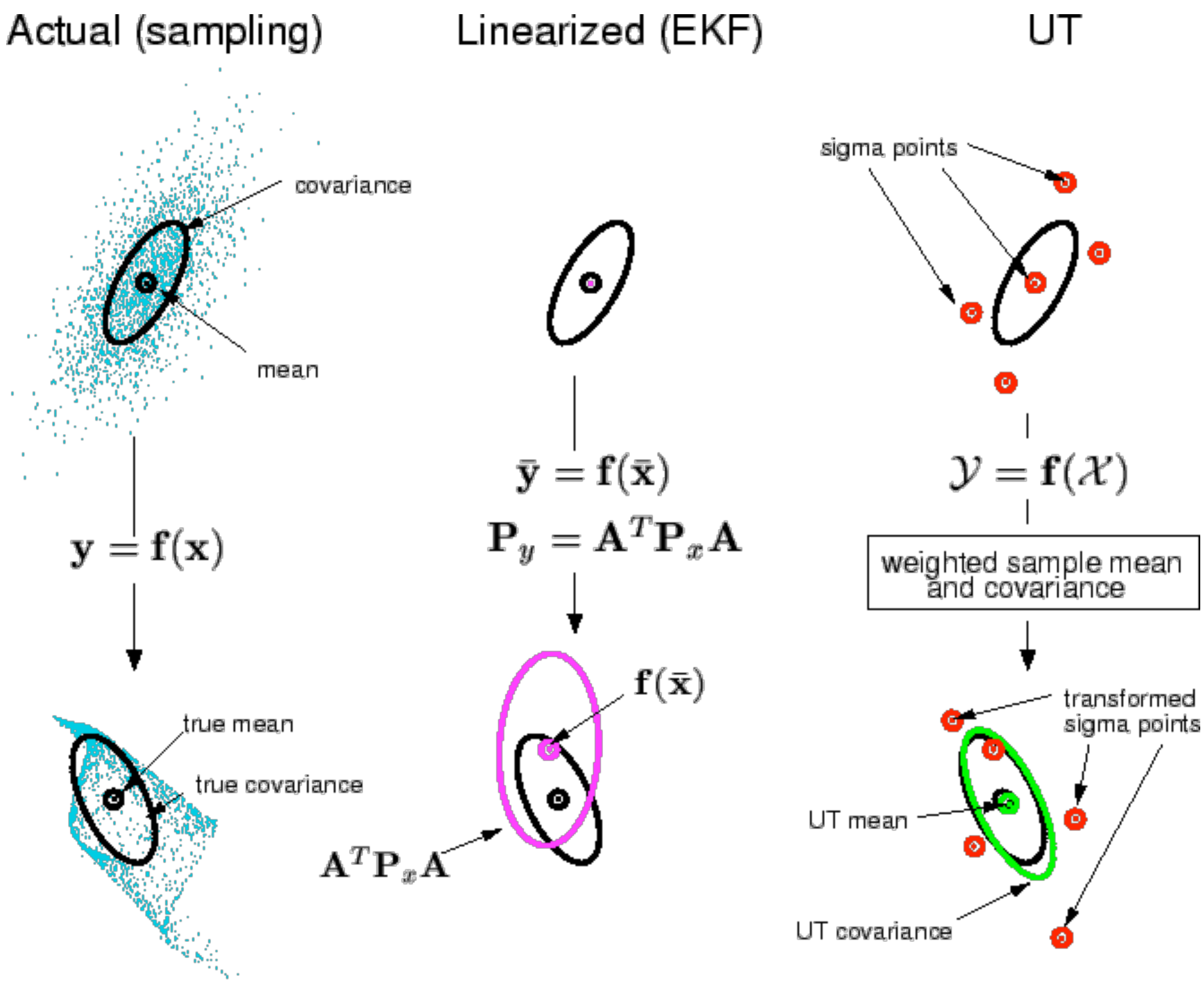
11: $K_t = \bar{\Sigma}_t^{x,z} \; S_t^{-1}$

12: $\mu_t = \bar{\mu}_t + K_t(z_t - \hat{z}_t)$

13: $\Sigma_t = \bar{\Sigma}_t - K_t \; S_t \; K_t^T$

14: *return* μ_t, Σ_t

EKF vs UKF



Courtesy: E.A. Wan and R. van der Merwe

UKF on SE3

- Easy to extend for manifolds:

$$\mathcal{N}(\mu, \Sigma) := \mu \boxplus \mathcal{N}(0, \Sigma),$$

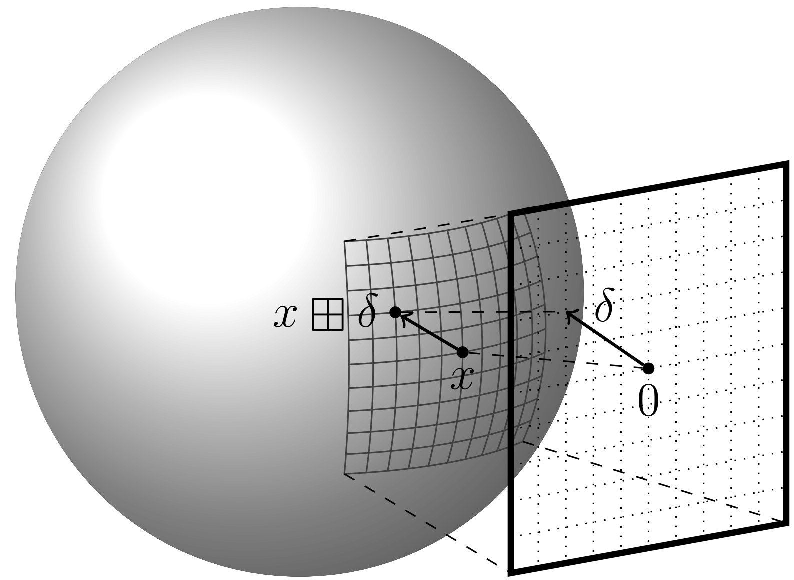
- Plus operator (expmap on SE3):

$$\boxplus : \mathcal{S} \times \mathbb{R}^n \to \mathcal{S}$$

- Minus operator (logmap on SE3):

$$\boxminus : \mathcal{S} \times \mathcal{S} \to \mathbb{R}^n.$$

- Works with small uncertainty



Unscented Transform on SE(3):
C. Hertzberg et al., "Integrating Generic Sensor Fusion Algorithms with Sound State Representations through Encapsulation of Manifolds", http://arxiv.org/pdf/1107.1119.pdf

Summary: State Estimation

- Probabilistic state estimation
  - Uncertainty in measurement and state-transition
  - Bayes filter

- Kalman filters
  - Linear KF for continuous Gaussian state variables and Gaussian model noise
  - Linear KF is optimal (if model is valid)
  - Extended KF: allow for non-linear measurement and state-transition models
  - Unscented KF: improve on linearization in EKF through unscented transform
  - Efficient filtering techniques

What we will cover today

- Introduction to vision-based state estimation and control

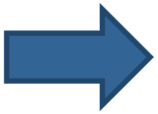
- State estimation
  - Bayes Filter
  - Extended Kalman Filter
  - Unscented Kalman Filter

- **Feedback Control**
  - **PID Control**
  - **Cascaded Control**

Feedback Control

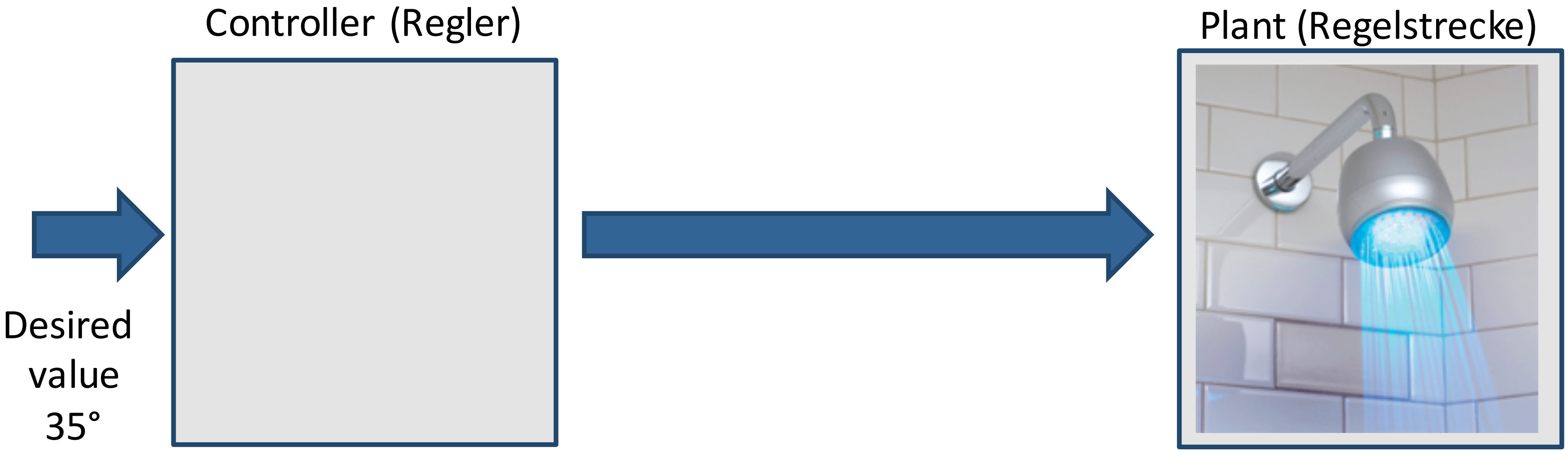
- Given:
  - Goal state x_{des}
  - Measured state (feedback) z
- Wanted:
  - Control signal u to reach goal state
- How to compute the control signal?

Feedback Control - Generic Idea

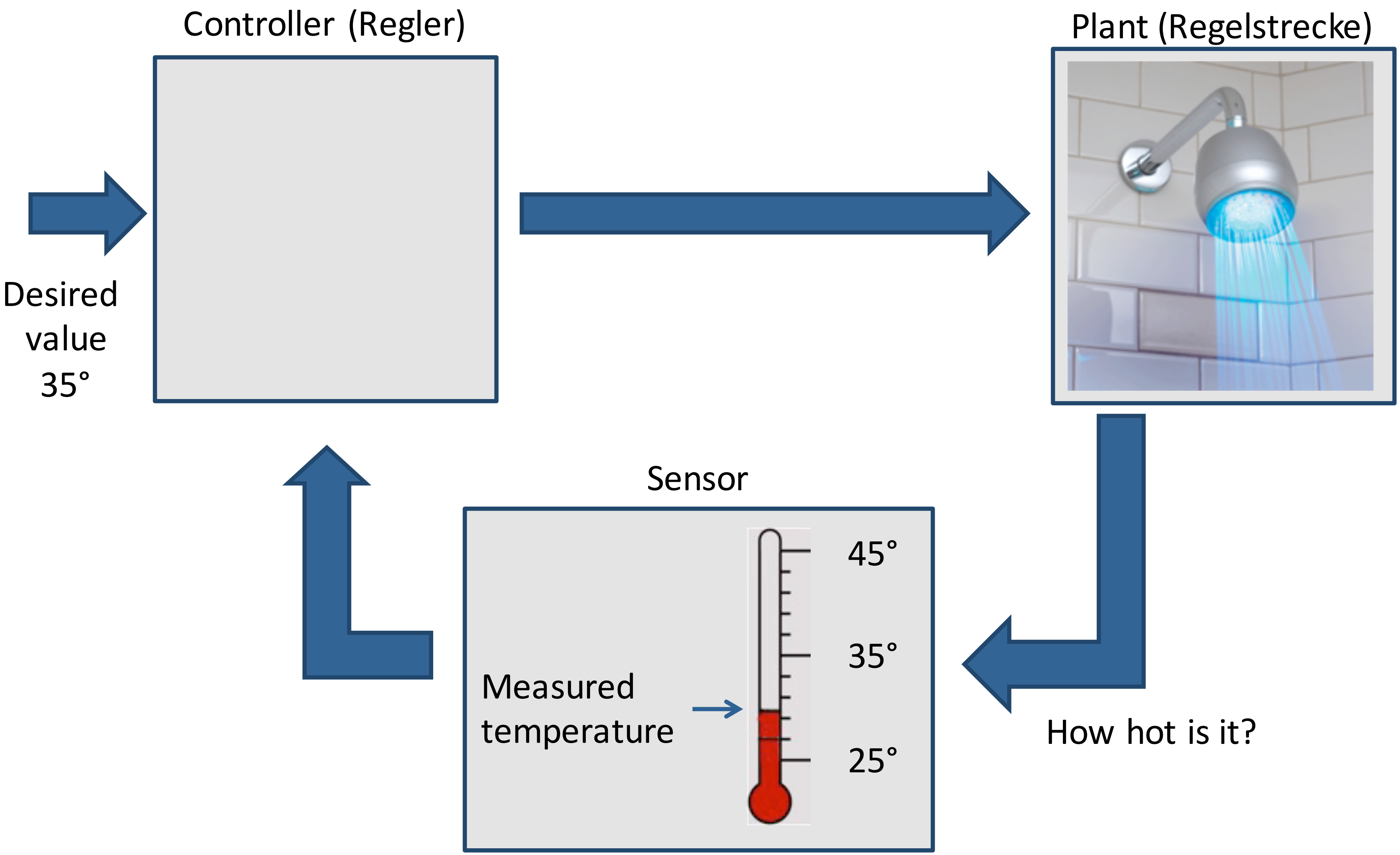


Desired
value
35°

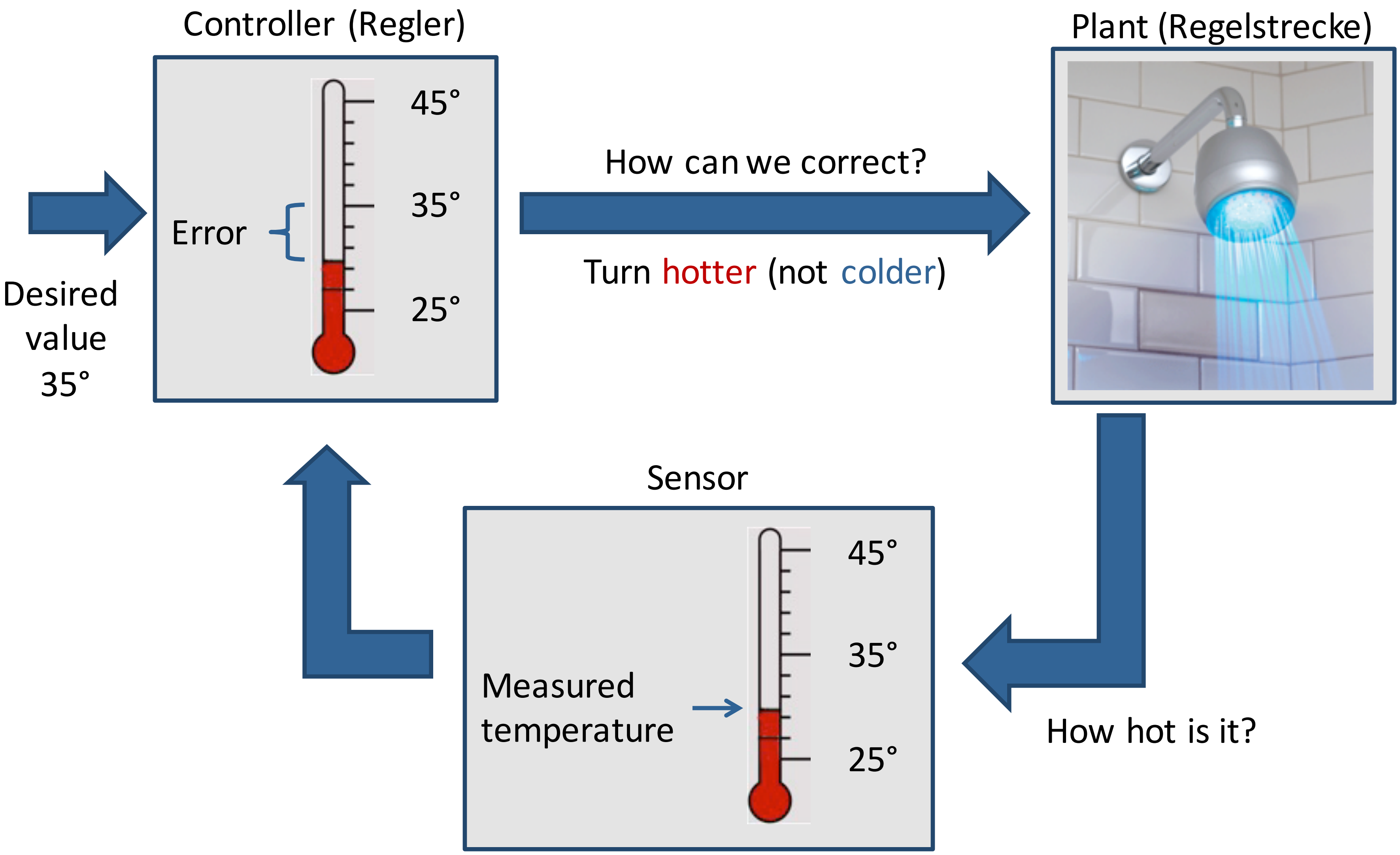
Feedback Control - Generic Idea



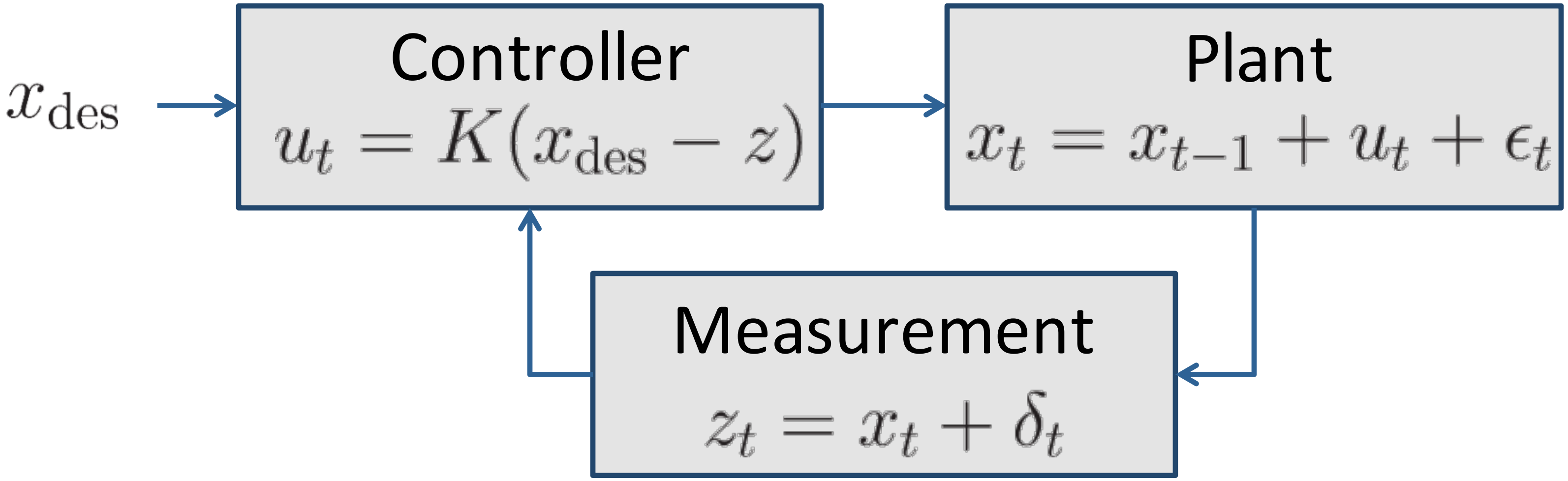
Feedback Control - Generic Idea

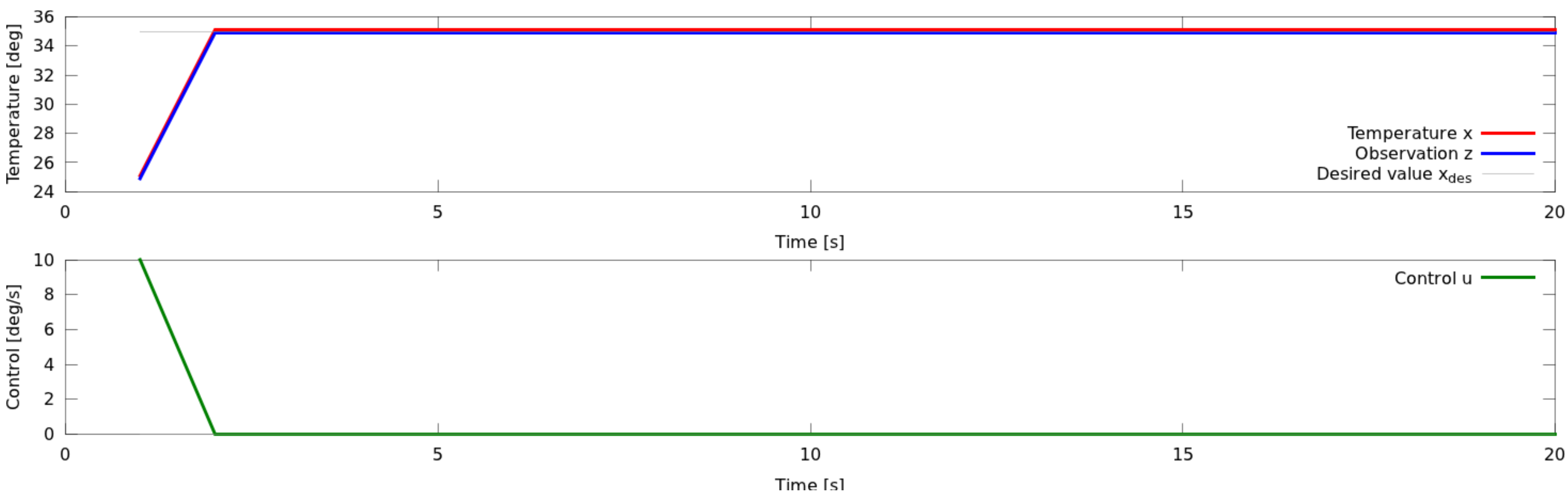


Feedback Control - Generic Idea



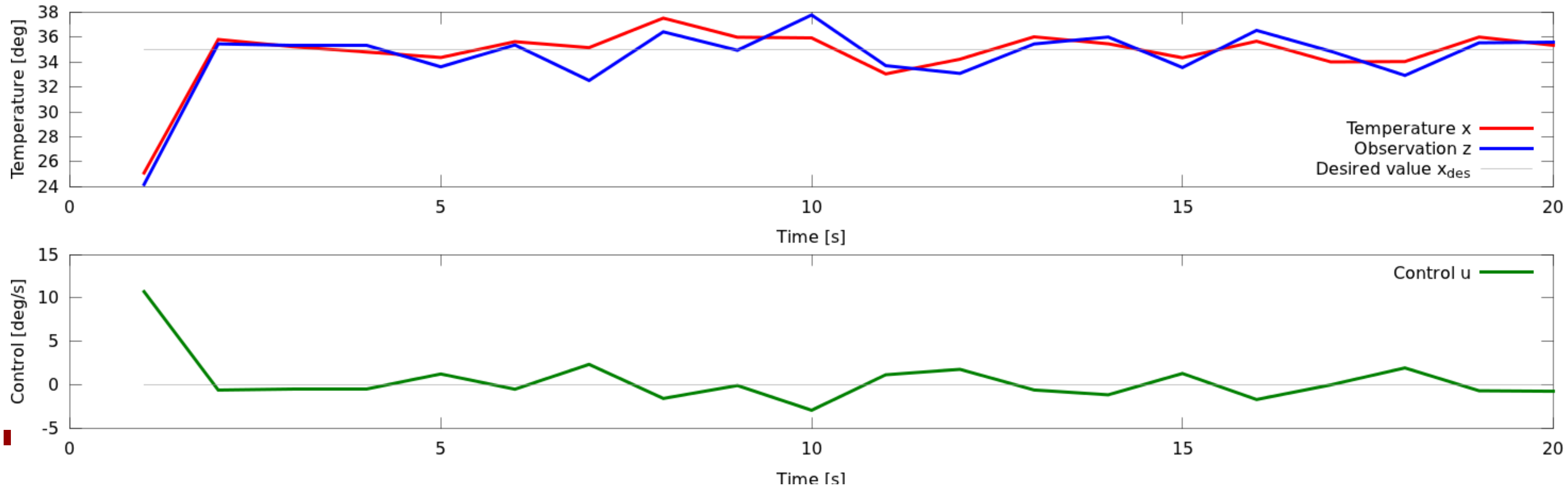
Feedback Control - Example





Measurement Noise

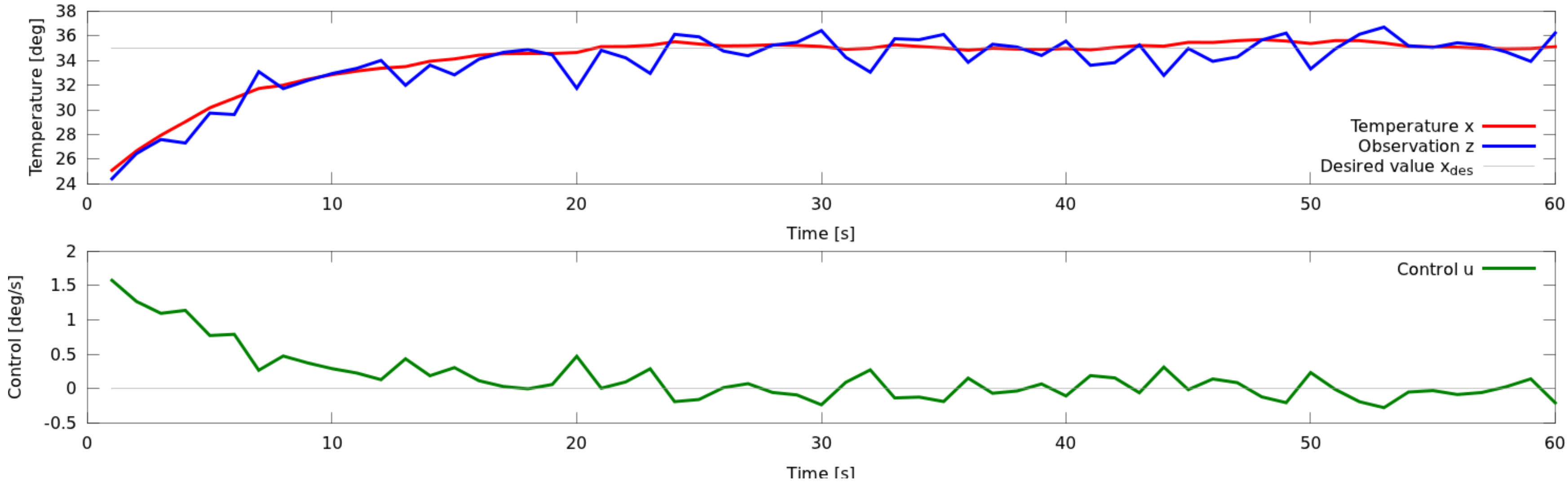
- What effect has noise in the measurements?



- How can we fix this?

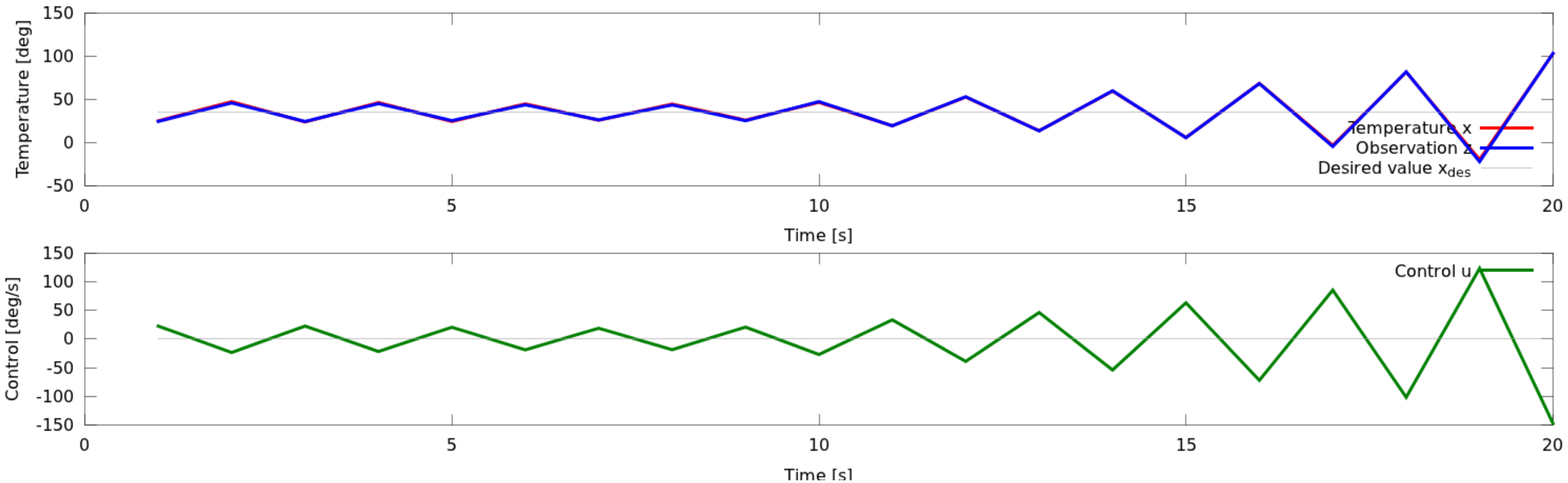
Proper Control with Measurement Noise

- Lower the gain… (K=0.15)



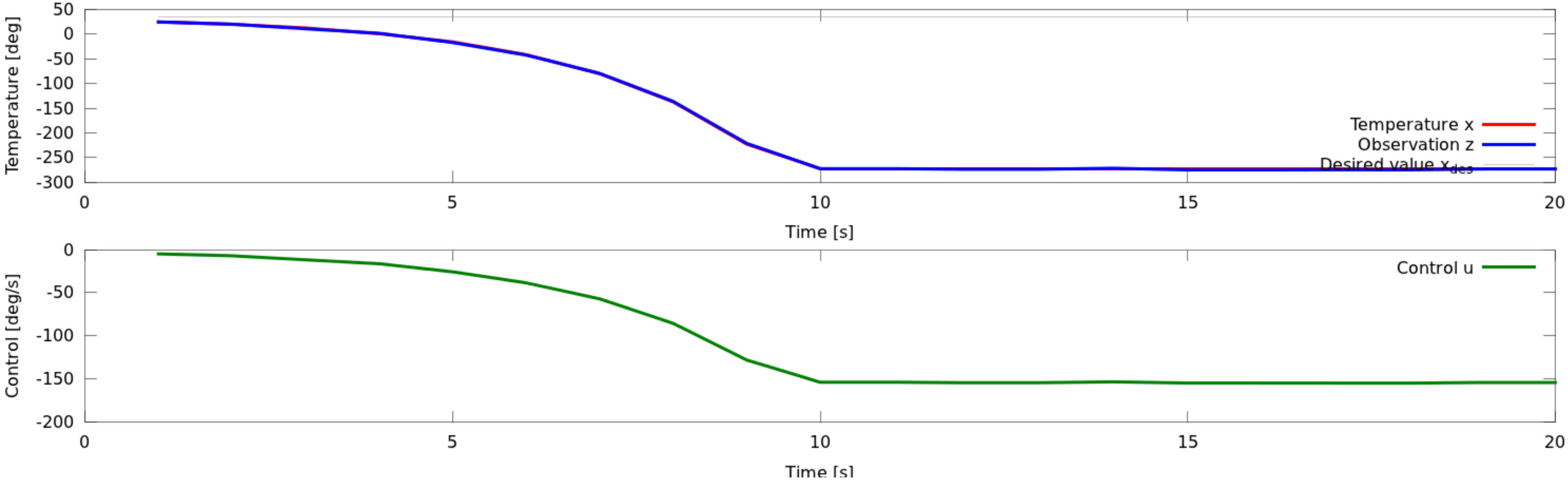
What do High Gains do?

- High gains are always problematic (K=2.15)



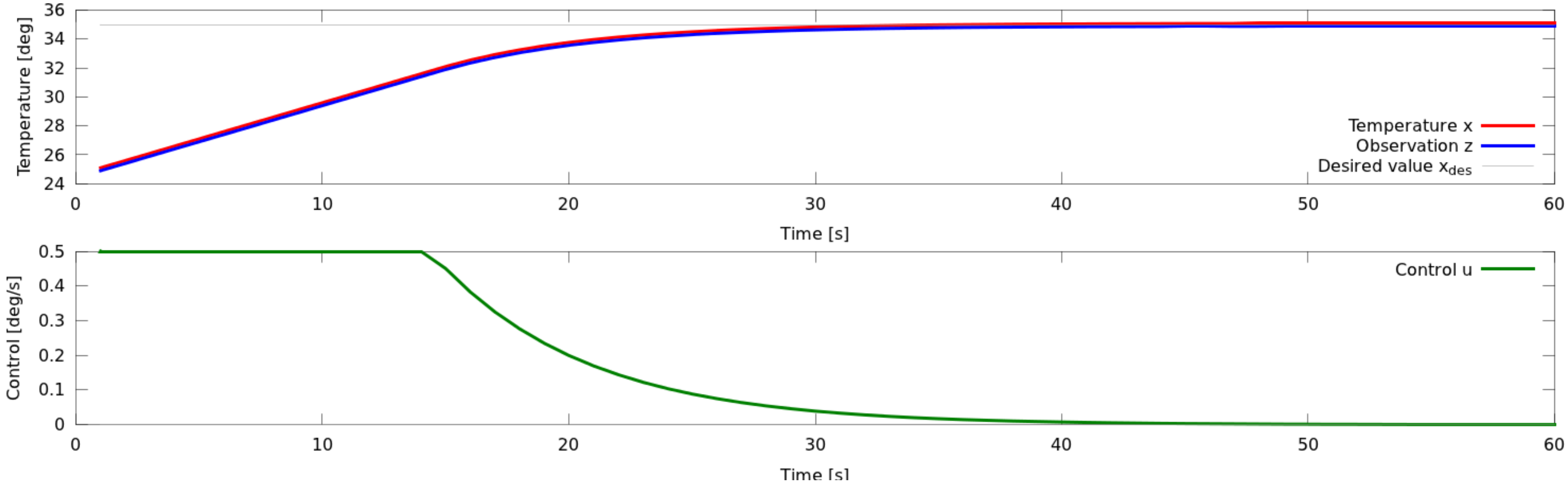
What happens if sign is messed up?

- Check K=-0.5

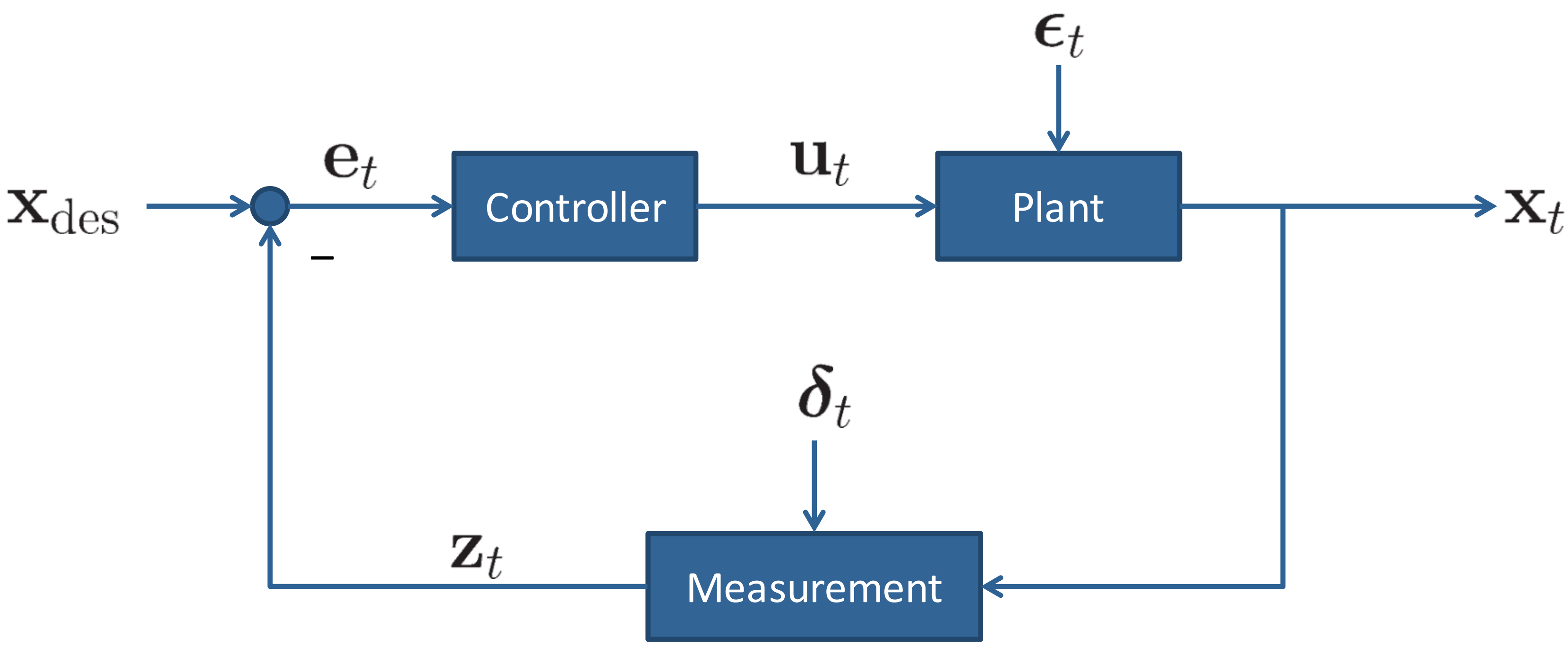


Saturation

- In practice, often the set of admissible controls u is bounded
- This is called (control) saturation

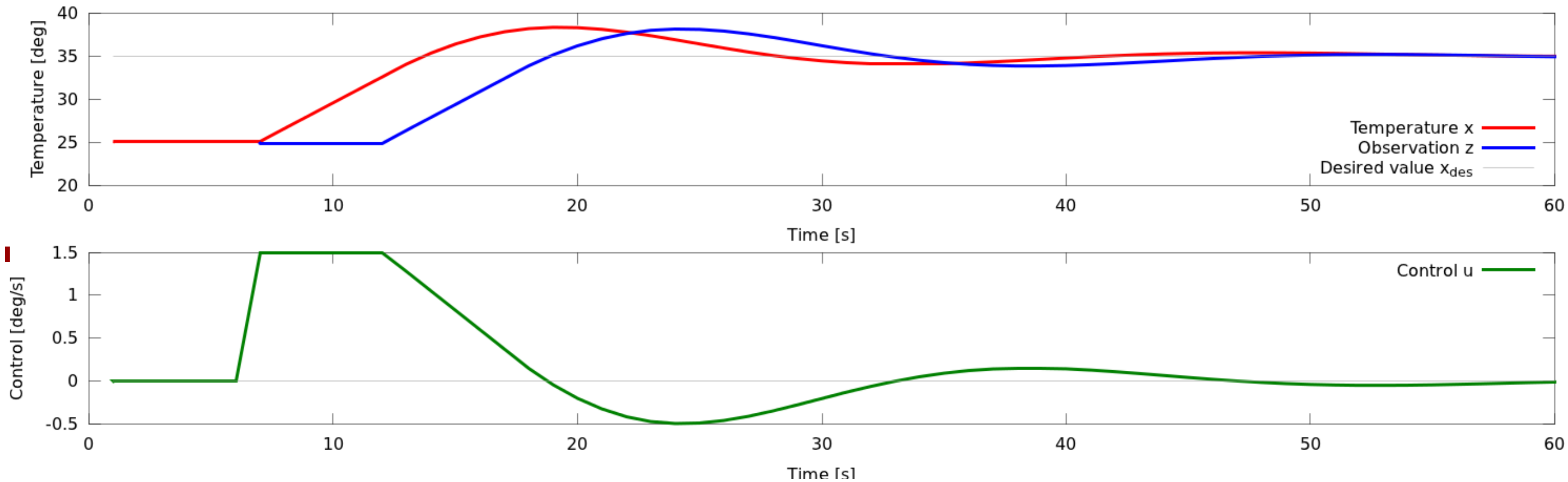


Block Diagram



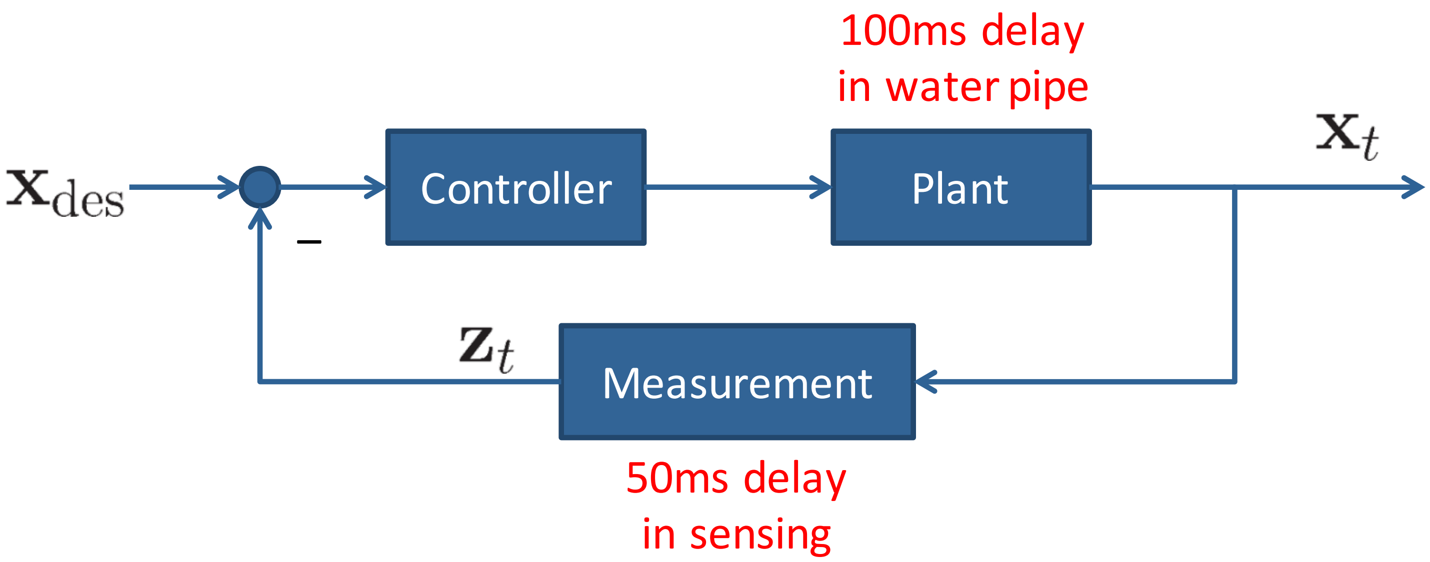
Delays

- In practice most systems have delays
- Can lead to overshoots/oscillations/de-stabilization



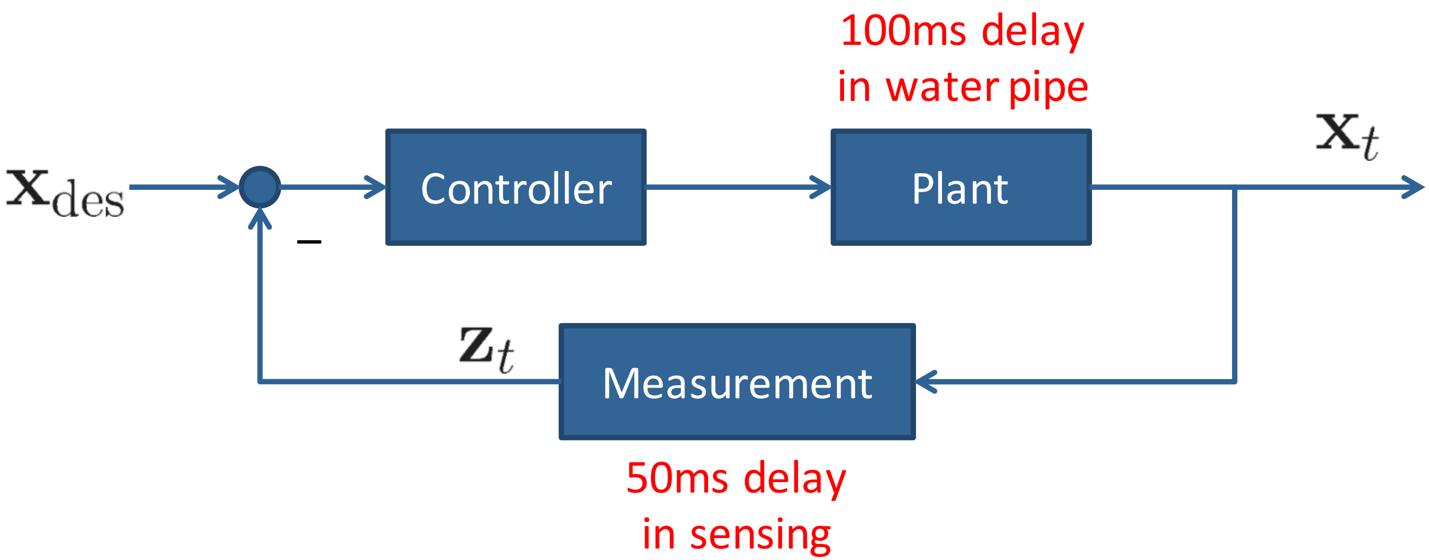
Delays

- What is the total dead time of this system?



Delays

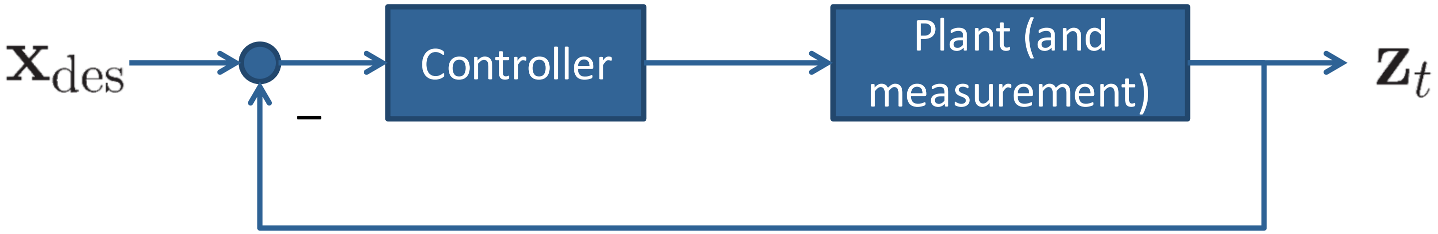
- What is the total dead time of this system?



- Can we distinguish delays in the measurement from delays in actuation?

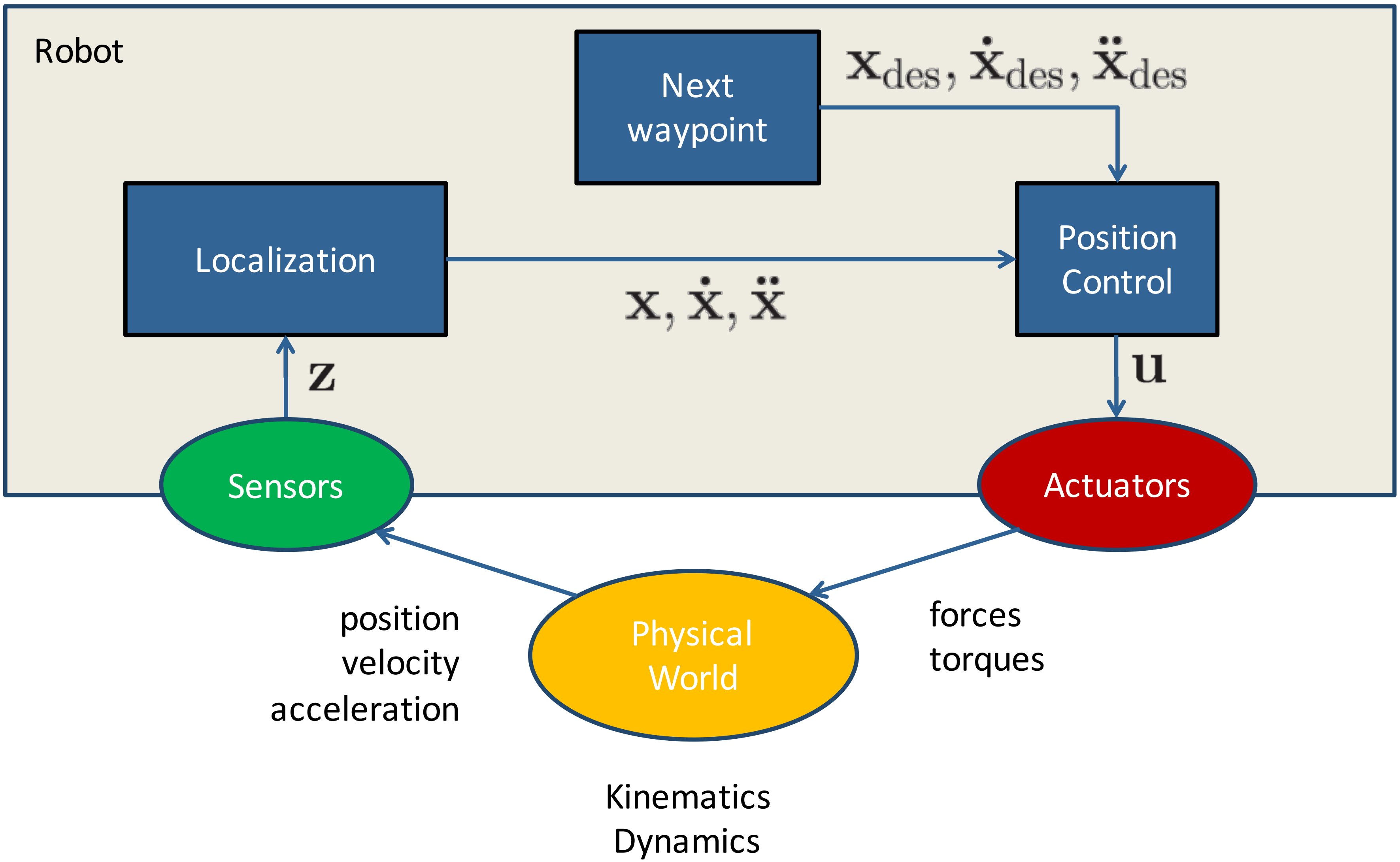
Delays

- What is the total dead time of this system?



- Can we distinguish delays in the measurement from delays in actuation? No!

Position Control



Rigid Body Kinematics

- Consider a rigid body
- Free floating in 1D space, no gravity
- In each time instant, we can apply a force F
- Results in acceleration $\ddot{x} = F/m$
- Desired position $x_{\text{des}} = 1$

P Control

- What happens for this control law?

$$u_t = K(x_{\text{des}} - x_{t-1})$$

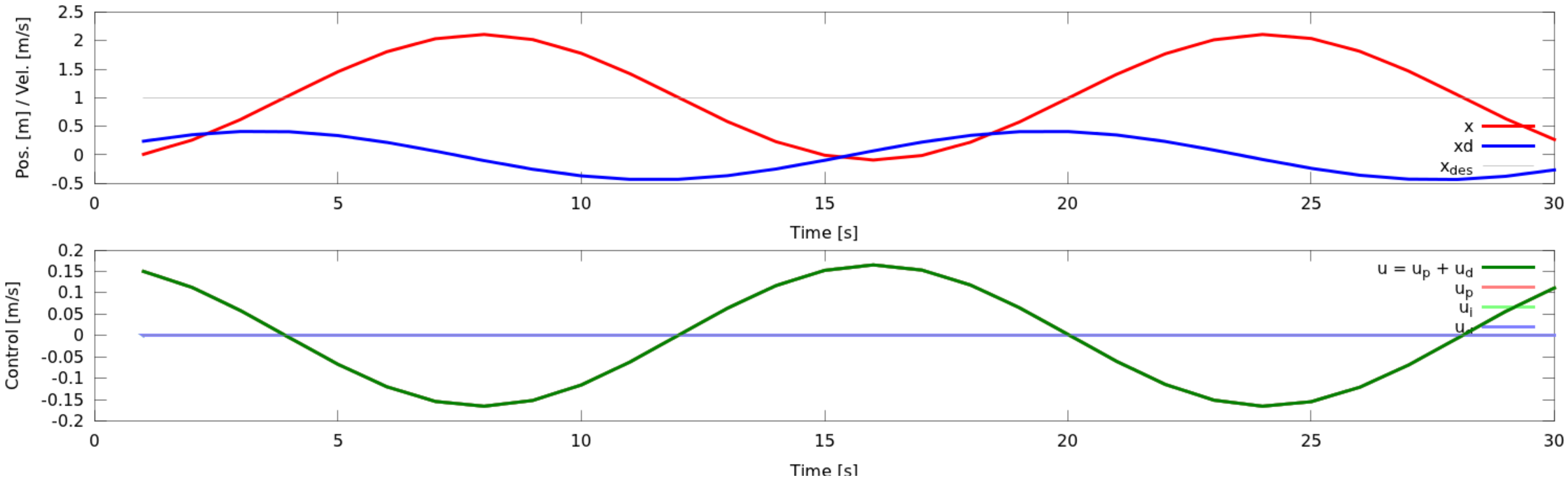
- This is called proportional control

P Control

- What happens for this control law?

$$u_t = K(x_{\text{des}} - x_{t-1})$$

- This is called proportional control

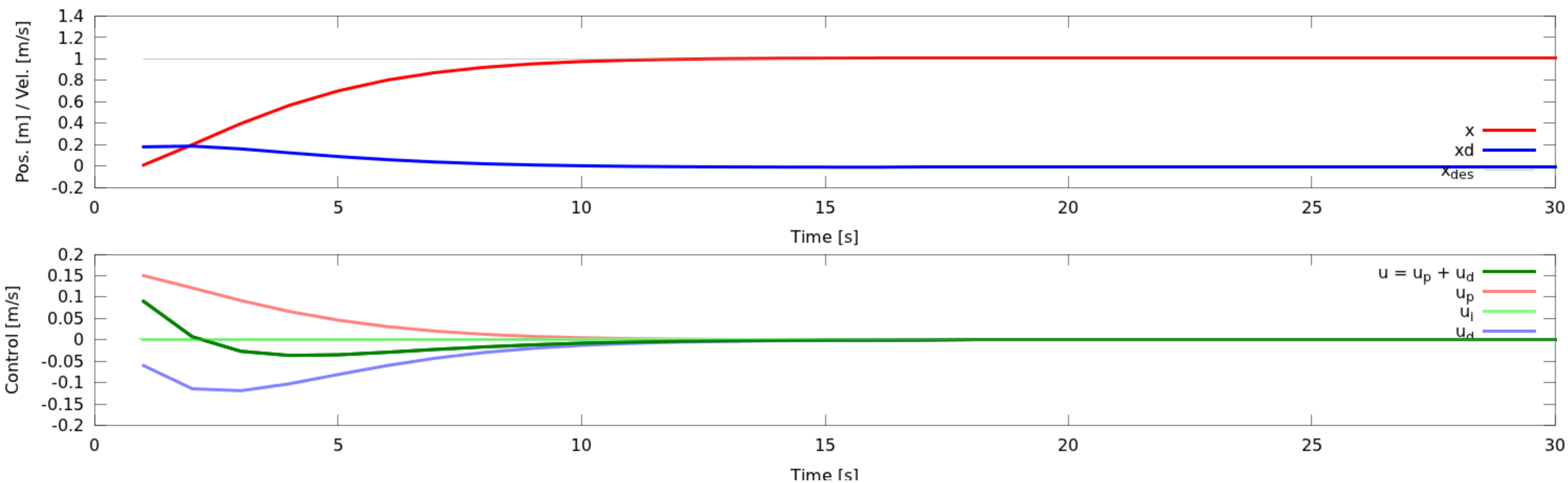


PD Control

- What happens for this control law?

$$u_t = K_P(x_{\text{des}} - x_{t-1}) + K_D(\dot{x}_{\text{des}} - \dot{x}_{t-1})$$

- Proportional-Derivative control

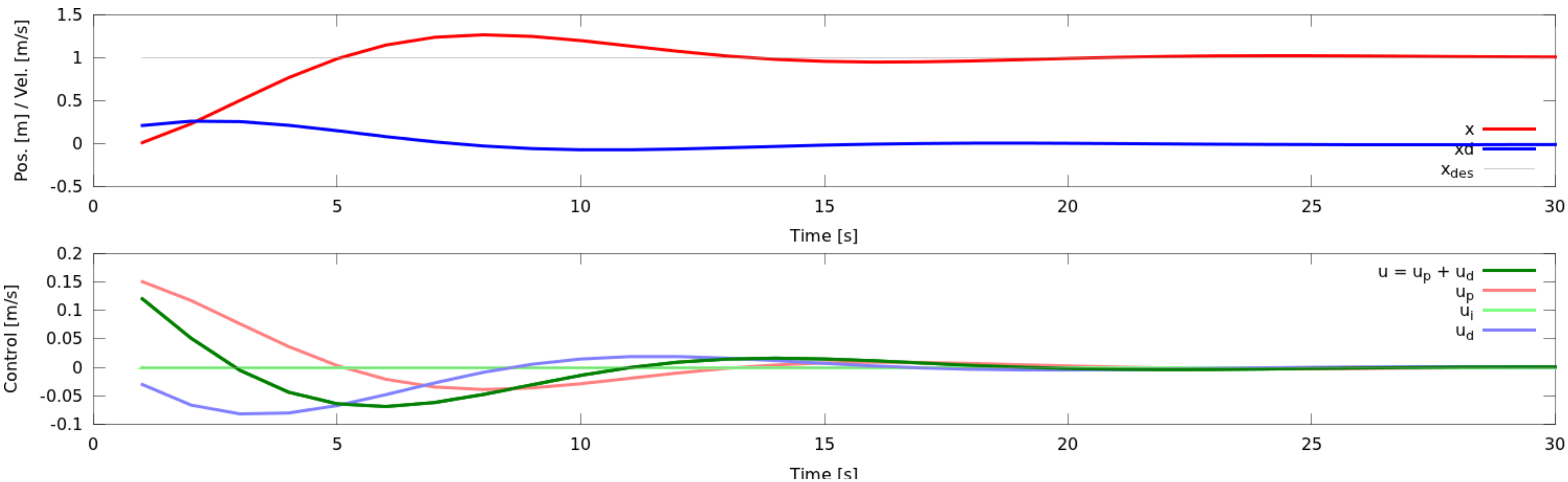


PD Control

- What happens for this control law?

$$u_t = K_P(x_{\text{des}} - x_{t-1}) + K_D(\dot{x}_{\text{des}} - \dot{x}_{t-1})$$

- What if we set **higher** gains?

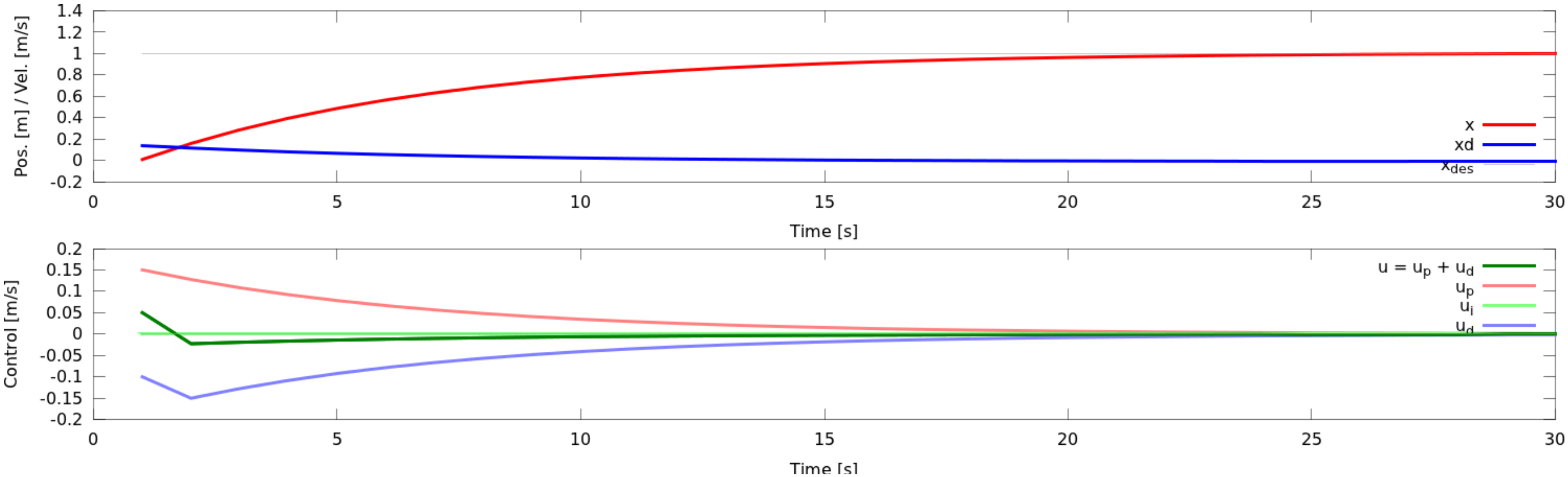


PD Control

- What happens for this control law?

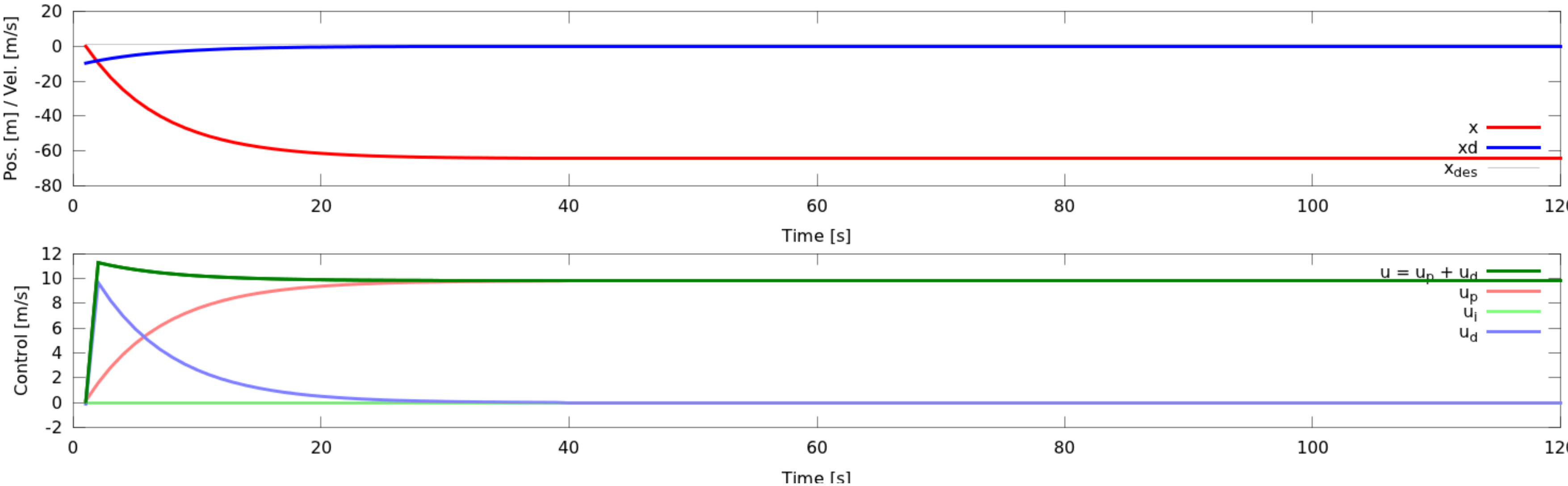
$$u_t = K_P(x_{\text{des}} - x_{t-1}) + K_D(\dot{x}_{\text{des}} - \dot{x}_{t-1})$$

- What if we set **lower** gains?



PD Control

- What happens when we add gravity?

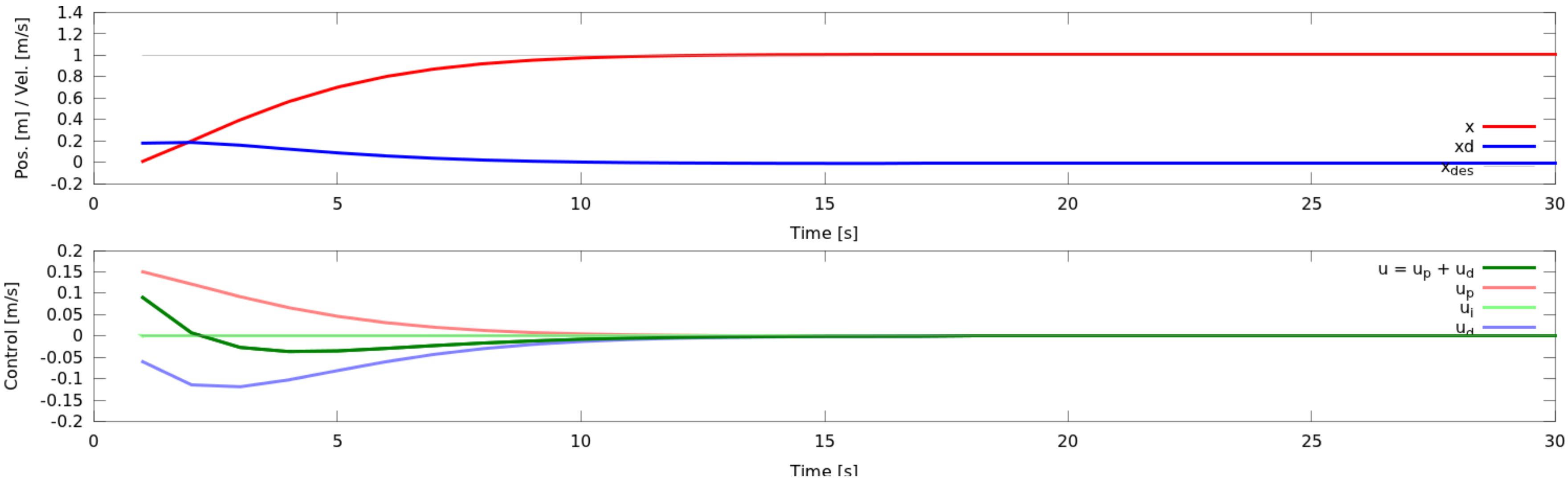


Gravity compensation

- Add as an additional term in the control law

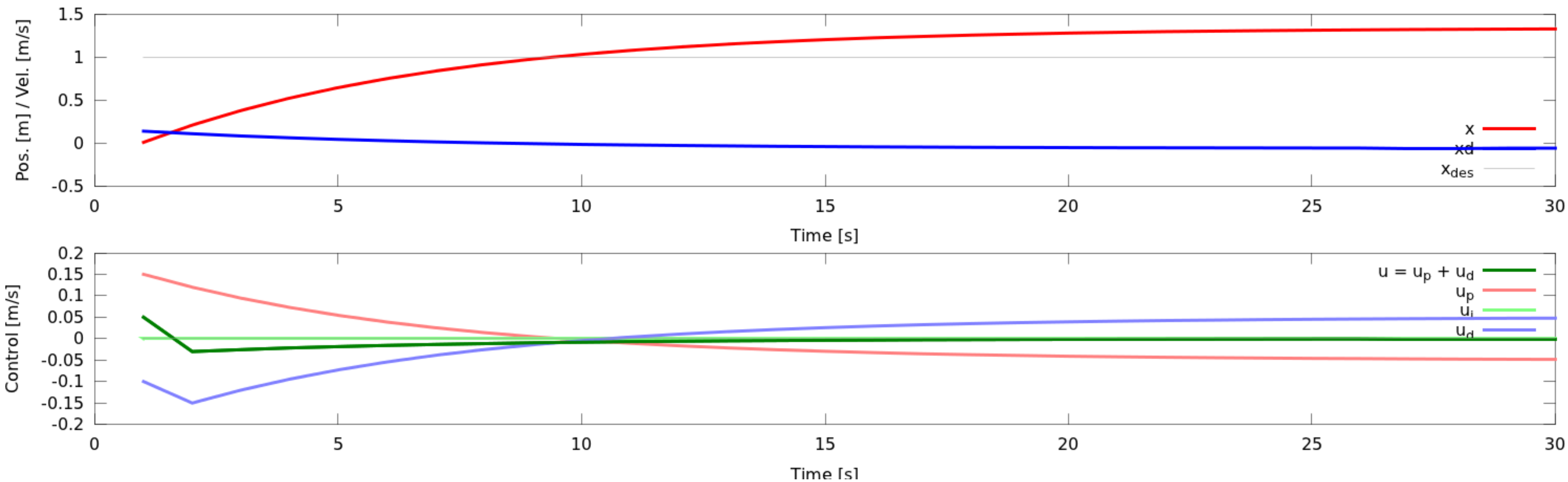
$$u_t = K_P(x_{\text{des}} - x_{t-1}) + K_D(\dot{x}_{\text{des}} - \dot{x}_{t-1}) + F_{\text{grav}}$$

- Any known (inverse) dynamics can be included



PD Control

- What happens when we have systematic errors? (control/sensor noise with non-zero mean)
- Example: unbalanced quadrocopter, wind, …
- Does the robot ever reach its desired location?

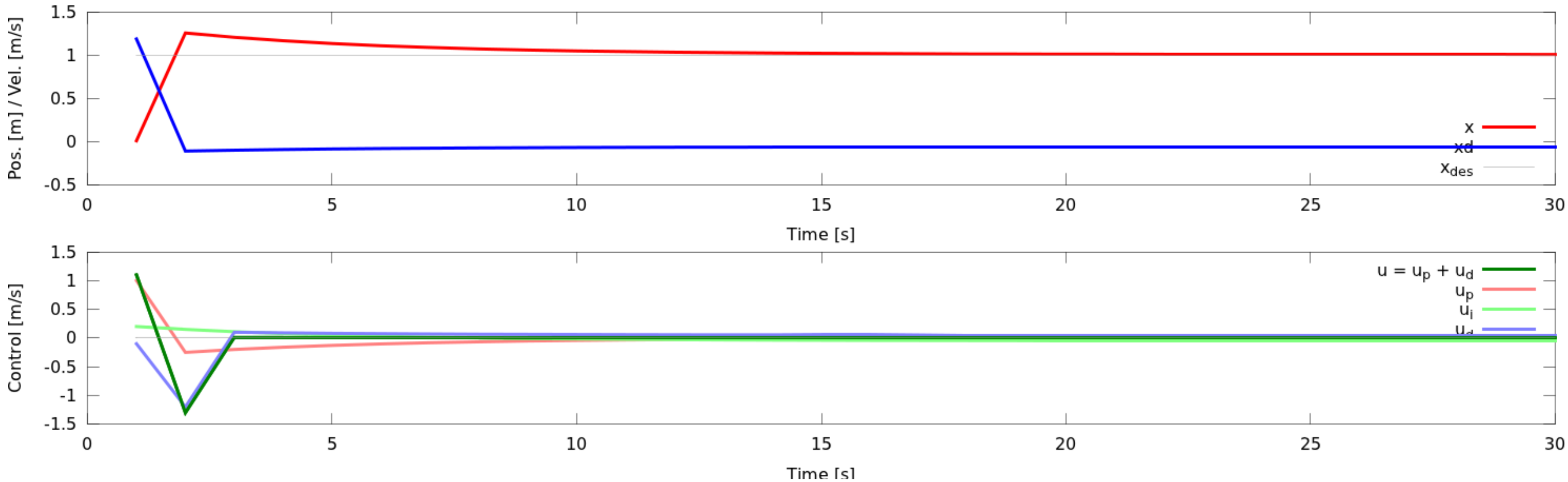


PID Control

- Idea: Estimate the system error (bias) by integrating the error

$$u_t = K_P(x_{\text{des}} - x_t) + K_D(\dot{x}_{\text{des}} - \dot{x}_t) + K_I \int_{-\infty}^{t} x_{des} - x_t \mathrm{d}t$$

- Proportional+Derivative+Integral Control



PID Control

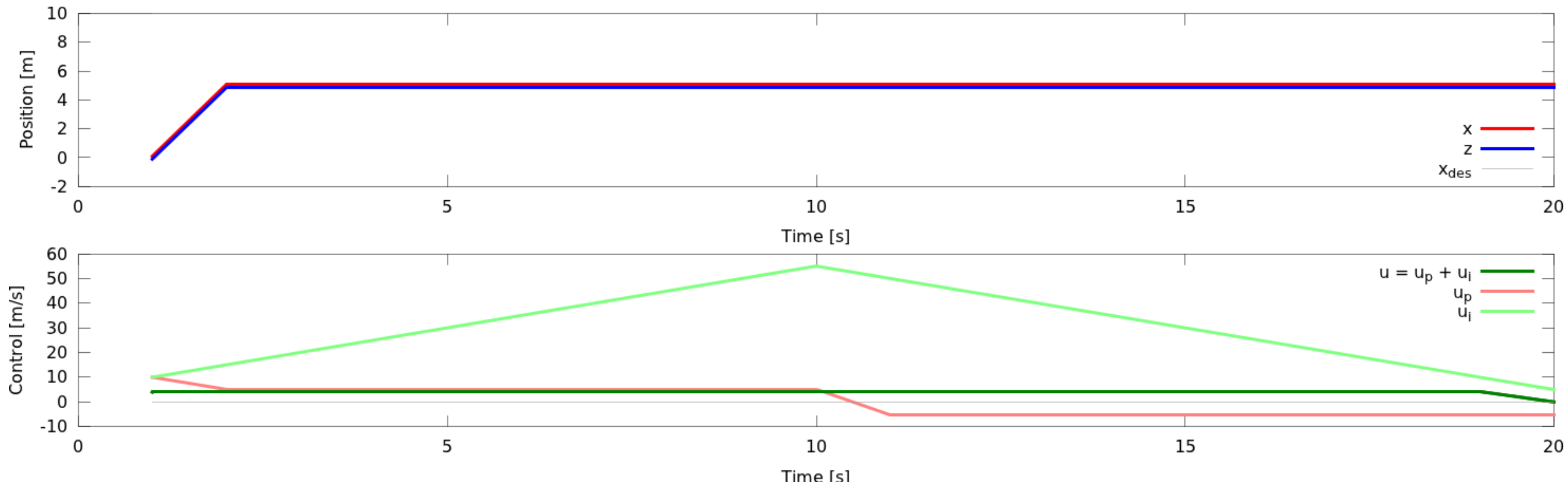
- Idea: Estimate the system error (bias) by integrating the error

$$u_t = K_P(x_{\text{des}} - x_t) + K_D(\dot{x}_{\text{des}} - \dot{x}_t) + K_I \int_{-\infty}^{t} x_{des} - x_t \mathrm{d}t$$

- Proportional+Derivative+Integral Control
- For steady state systems, this can be reasonable
- Otherwise, it may create havoc or even disaster (wind-up effect)

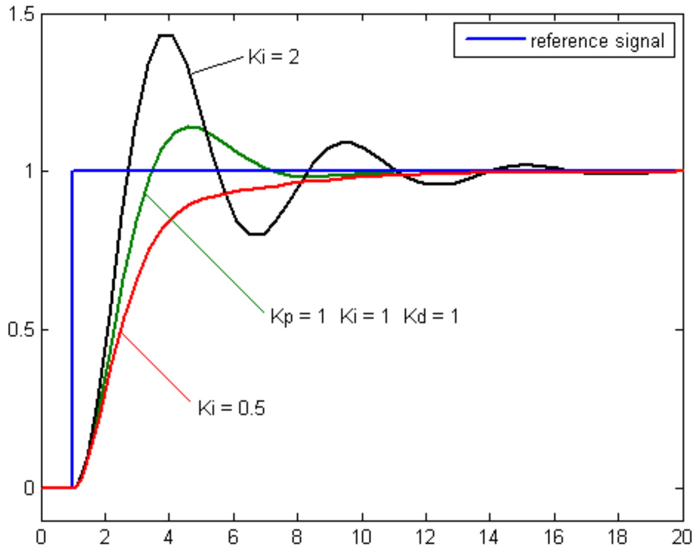
Example: Wind-up effect

- Quadrocopter gets stuck in a tree → does not reach steady state
- What is the effect on the I-term?



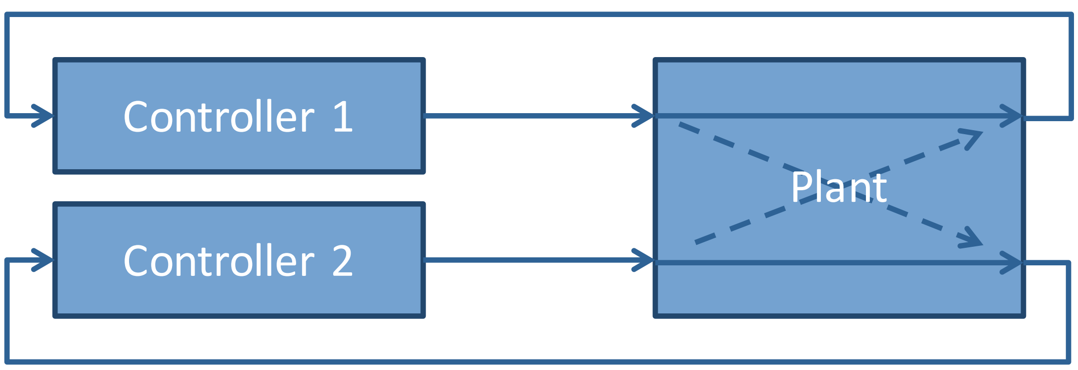
How to Choose the Coefficients?

- Gains too large: overshooting, oscillations
- Gains too small: long time to converge
- Heuristic methods exist
- In practice, often tuned manually

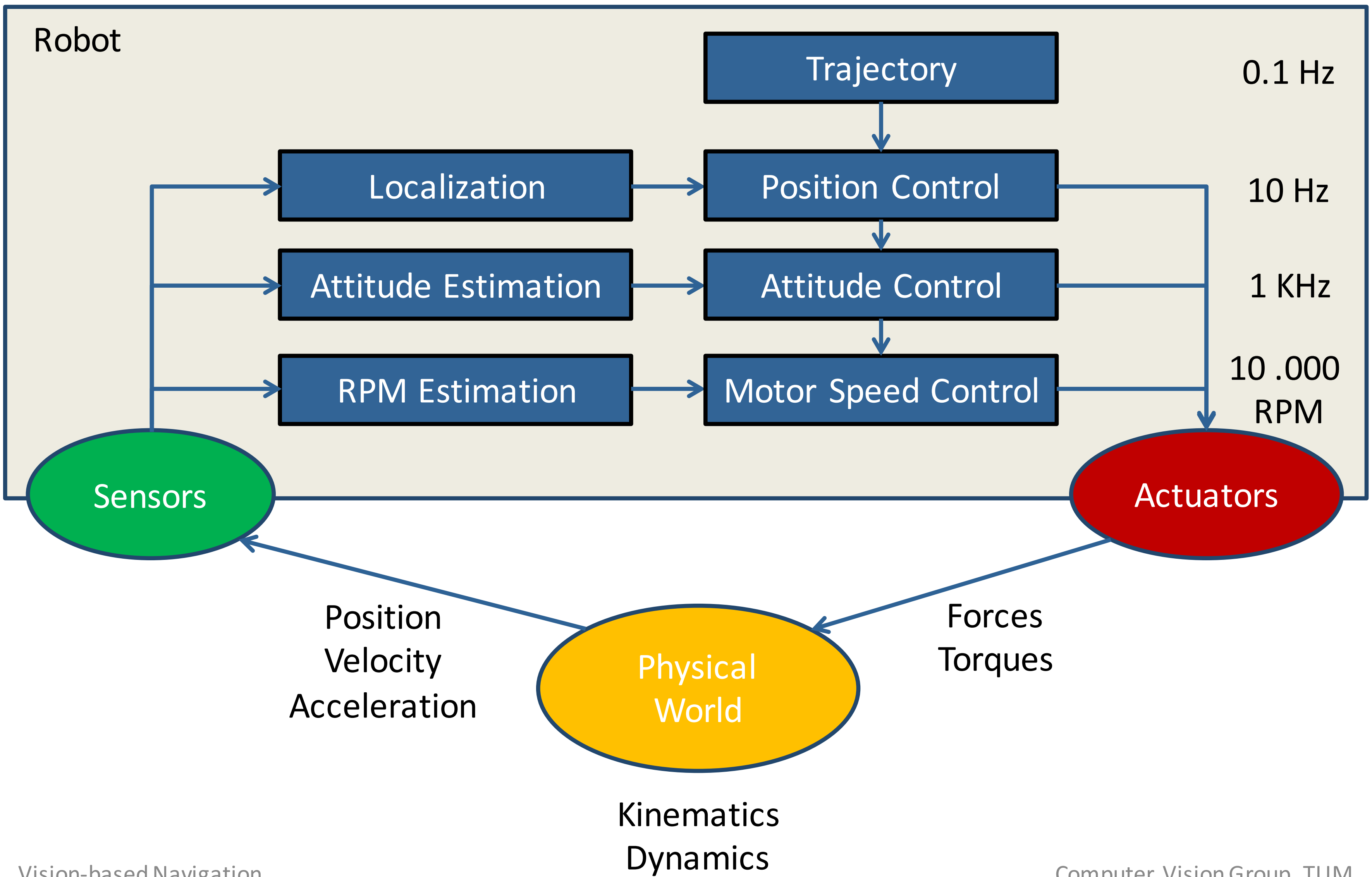


De-coupled Control

- So far, we considered only single-input, single-output systems (SISO)
- Real systems have multiple inputs + outputs
- MIMO (multiple-input, multiple-output)
- In practice, control is often de-coupled



Cascaded Control



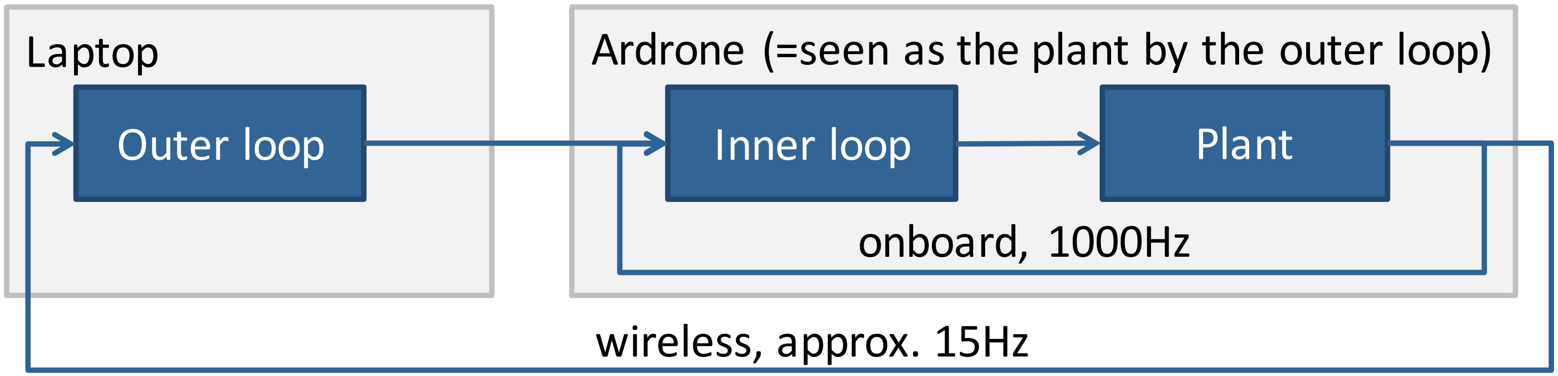
Assumptions of Cascaded Control

- Dynamics of inner loops is so fast that it is not visible from outer loops
- Dynamics of outer loops is so slow that it appears as static to the inner loops

Example: Ardrone

Cascaded control

- Inner loop runs on embedded PC and stabilizes flight
- Outer loop runs externally and implements position control



Ardrone: Inner Control Loop

- Plant input: motor torques

$$\mathbf{u}_{\text{inner}} = \begin{pmatrix} \tau_1 & \tau_2 & \tau_3 & \tau_4 \end{pmatrix}^\top$$

- Plant output: roll, pitch, yaw rate, z velocity

$$\mathbf{x}_{\text{inner}} = \begin{pmatrix} \underline{\omega_x \quad \omega_y \quad \omega_z} & \underline{z} \end{pmatrix}^\top$$

attitude
(measured using gyro +
accelerometer)

altitude
(measured using ultrasonic
distance sensor + attitude)

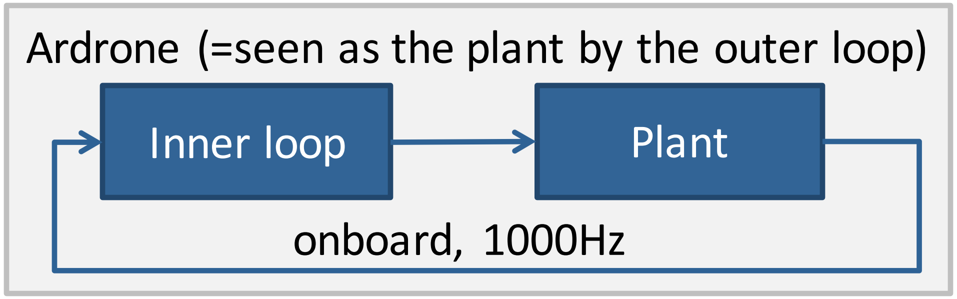
Ardrone: Inner Control Loop

- Plant input: motor torques

$$\mathbf{u}_{\text{inner}} = \begin{pmatrix}\tau_1 & \tau_2 & \tau_3 & \tau_4\end{pmatrix}^\top$$

- Plant output: roll, pitch, yaw rate, z velocity

$$\mathbf{x}_{\text{inner}} = \begin{pmatrix}\omega_x & \omega_y & \omega_z & z\end{pmatrix}^\top$$

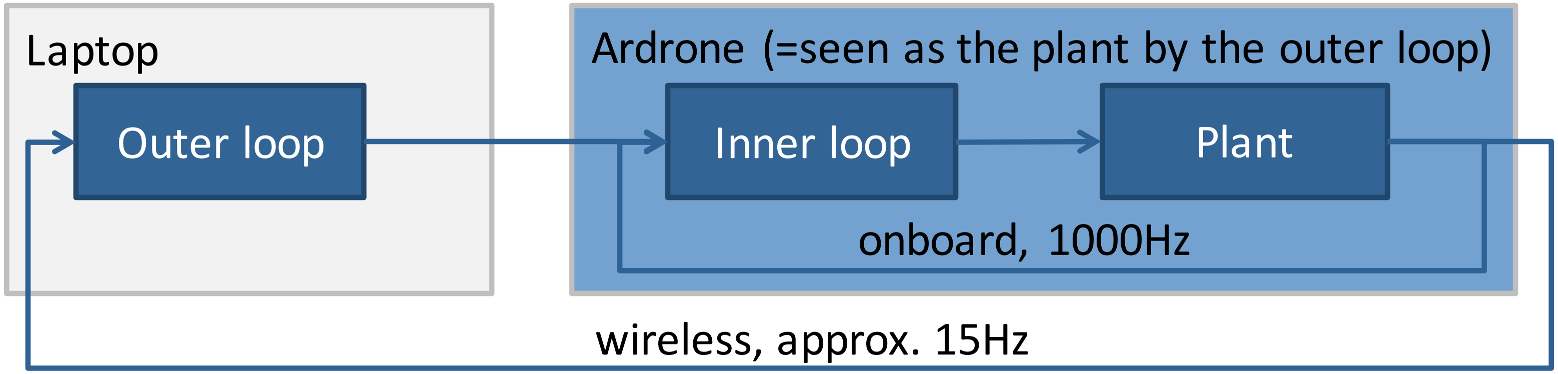


Ardrone: Outer Control Loop

- Outer loop sees inner loop as a plant (black box)
- Plant input: roll, pitch, yaw rate, z velocity
- Plant output:

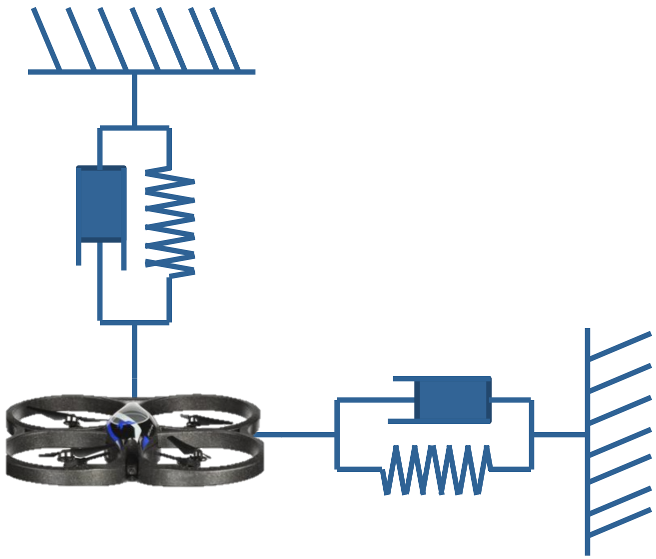
$$\mathbf{u}_{\text{outer}} = \begin{pmatrix}\omega_x & \omega_y & \dot{\omega}_z & \dot{z}\end{pmatrix}^\top$$

$$\mathbf{x}_{\text{outer}} = \begin{pmatrix}x & y & z & \psi\end{pmatrix}^\top$$



Mechanical Equivalent

- PD Control is equivalent to adding spring-dampers between the desired values and the current position



Advanced Control Techniques

What other control techniques do exist?

- Adaptive control
- Robust control
- Optimal control
- Linear-quadratic regulator (LQR)
- Reinforcement learning
- Inverse reinforcement learning
- ... and many more

Summary: Feedback Control

PID control is the most used control technique in practice

- P control → simple proportional control, often enough
- PI control → can compensate for bias (e.g., wind)
- PD control → can be used to reduce overshoot (e.g., when acceleration is controlled)
- PID control → all of the above

Lessons Learned Today

- Probabilistic state estimation techniques
  - Linear Kalman Filter, Extended KF, Unscented KF
  - Efficient filtering techniques, well suited for onboard processing
- How to control a system using PID controllers
  - Intuitive control laws
  - Easy to implement
  - Can be tricky to optimize parameters
- System simplifications: Decoupled and cascaded control

Questions ?