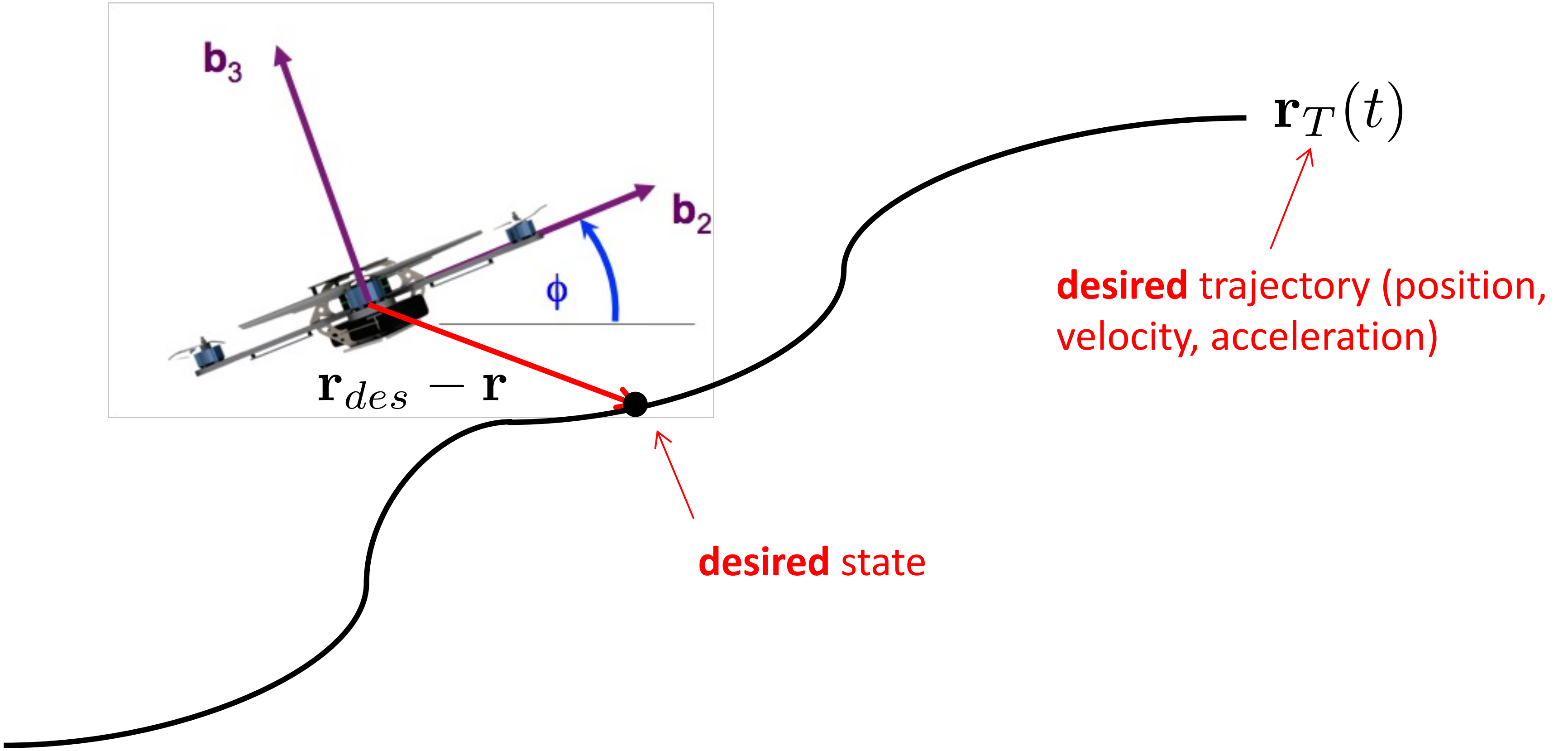
Feedback Control

Vijay Kumar and James Paulos

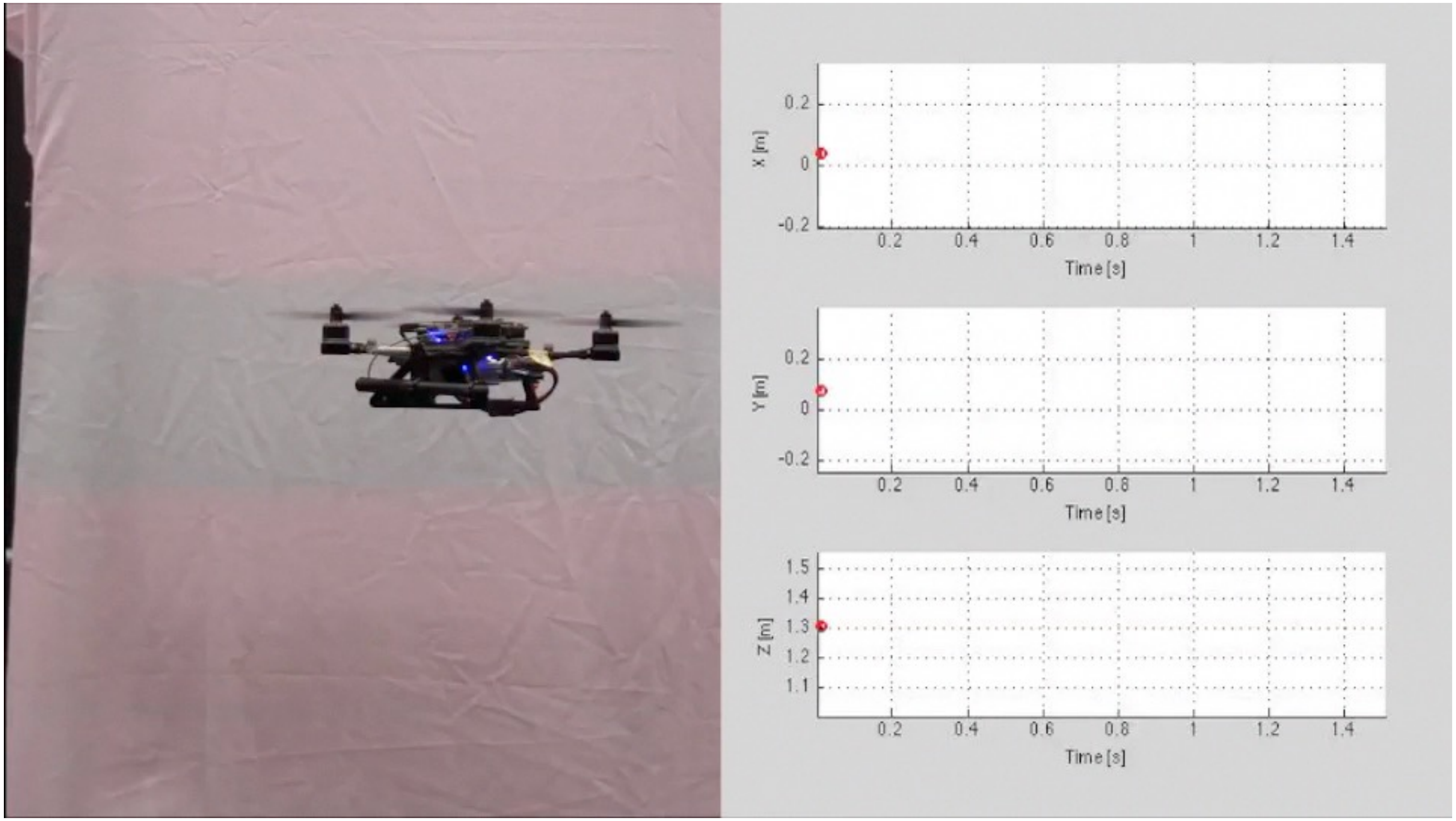
ENGINEERING DESIRABLE SYSTEM DYNAMICS

Penn Engineering | GRASP Laboratory
General Robotics, Automation, Sensing & Perception Lab

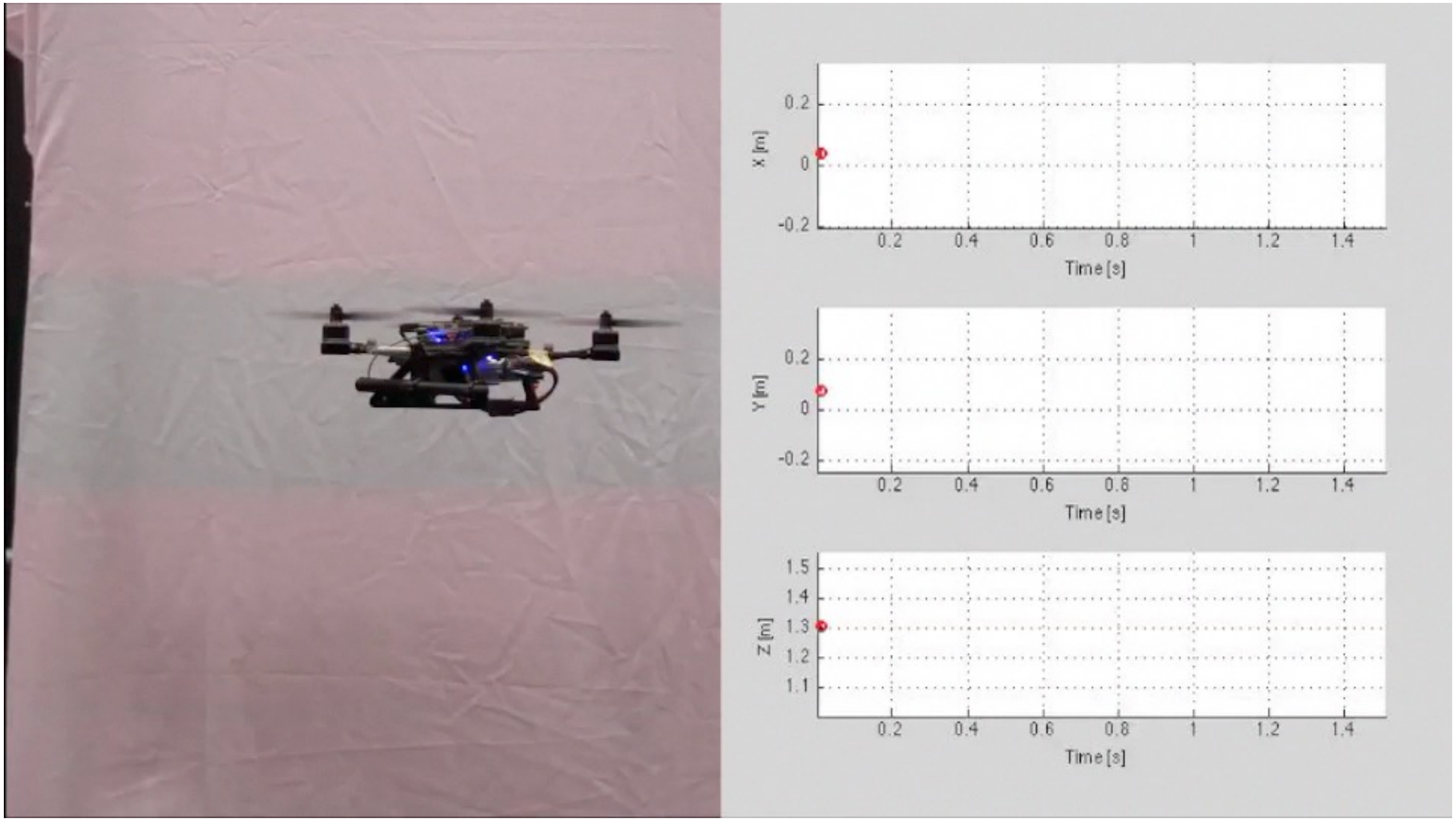
Trajectory Controller



Equilibrium.



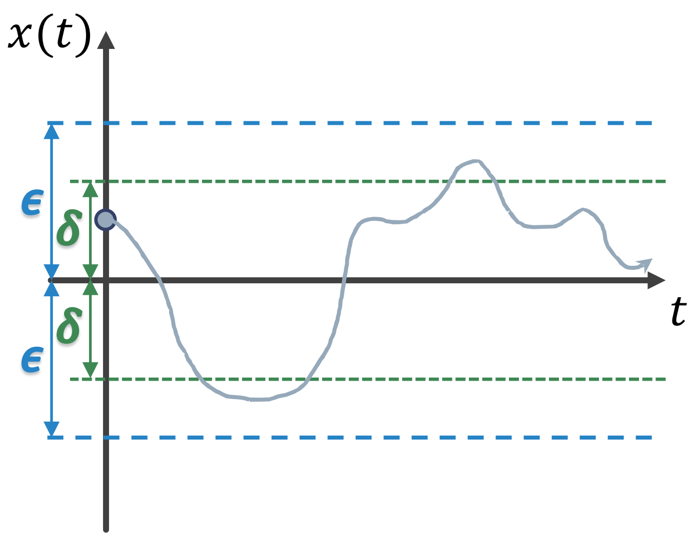
Stability?

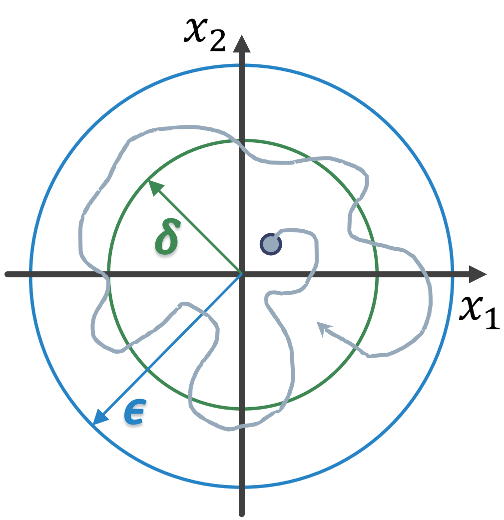


X [m]
0.2
0
-0.2
0.2 0.4 0.6 0.8 1 1.2 1.4
Time [s]
Y [m]
0.2
0
-0.2
0.2 0.4 0.6 0.8 1 1.2 1.4
Time [s]
Z [m]
1.5
1.4
1.3
1.2
1.1
0.2 0.4 0.6 0.8 1 1.2 1.4
Time [s]

Stability in the Sense of Lyapunov "Stability i.s.L."

An equilibrium point $\mathbf{x}_e$ of the system $\dot{\mathbf{x}} = f(\mathbf{x})$ is ***stable*** in the sense of Lyapunov if for any $\epsilon > 0$, there exists a value $\delta(t_0, \epsilon) > 0$ such that if $\|\mathbf{x}(t_0, \mathbf{x}_0) - \mathbf{x}_e\| < \delta$ then $\|\mathbf{x}(t; t_0, \mathbf{x}_0) - \mathbf{x}_e\| < \epsilon$ for all $t \geq t_0$.





- An equilibrium point is unstable if it is not stable i.s.L.
- The equilibrium point is ***uniformly stable*** i.s.L. if $\delta = \delta(\epsilon)$.

Stability i.s.L. is Weak just by Itself

- Stability i.s.L. means that the system state will remain close to the equilibrium point.
- Stability i.s.L. bounds how much the system state will fluctuate around the equilibrium point.

Does not answer…

- Will it ever reach the equilibrium point?
- Will it stay at the equilibrium point for all future times?

Asymptotic Stability

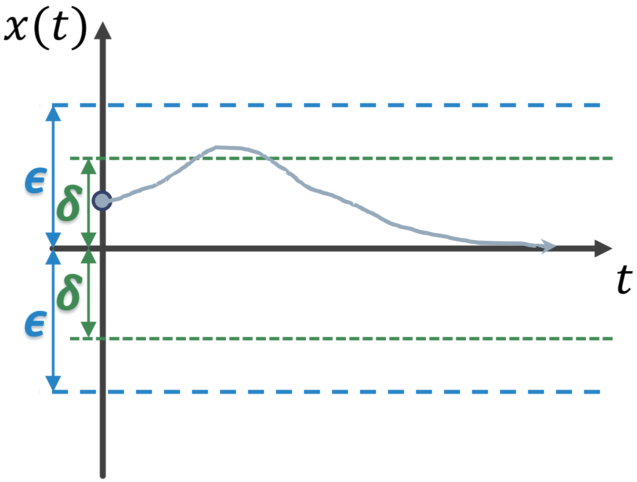
➢ An equilibrium point is asymptotically stable i.s.L. if it is:

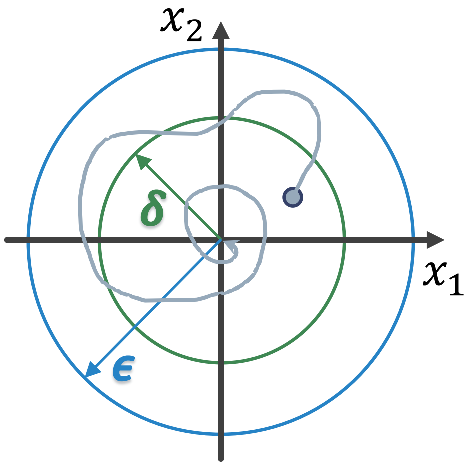
1. Stable (i.s.L.)

For any $\epsilon > 0$, there exists a value $\delta(t_0, \epsilon) > 0$ such that if $\|\mathbf{x}(t_0, \mathbf{x}_0) - \mathbf{x}_e\| < \delta$ then $\|\mathbf{x}(t; t_0, \mathbf{x}_0) - \mathbf{x}_e\| < \epsilon$ for all $t \geq t_0$.

2. Convergent

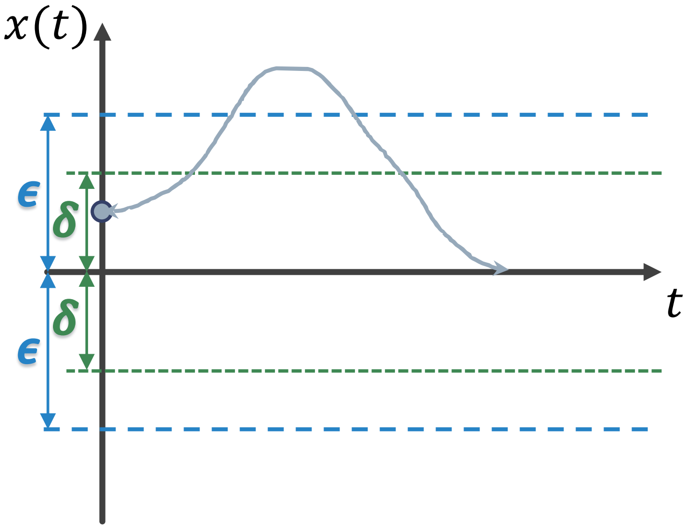
$\mathbf{x}(t; t_0, \mathbf{x}_0) \rightarrow \mathbf{x}_e$ as $t \rightarrow \infty$.



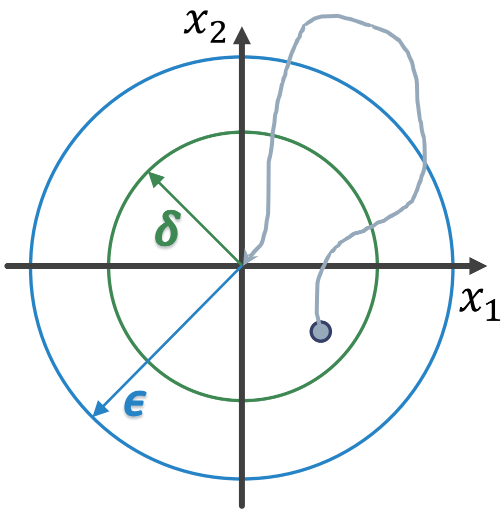


Asymptotic Stability

- **Note:** Convergence alone does not necessarily imply (asymptotic) stability! Why?



$x_e = 0$



$\mathbf{x}_e = [0 \quad 0]^{\mathrm{T}}$

Still does not answer…

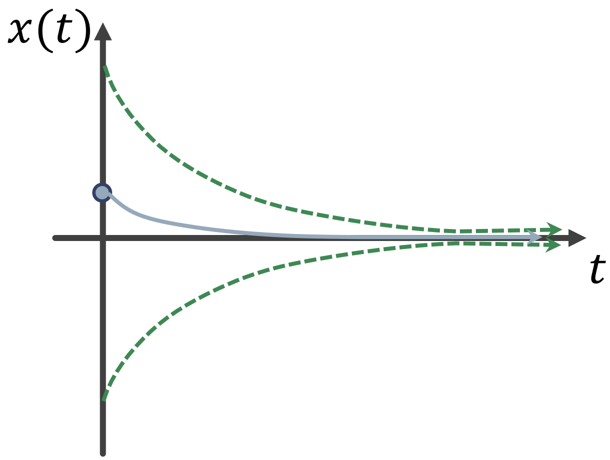
- How fast does it converge?

Exponential Stability

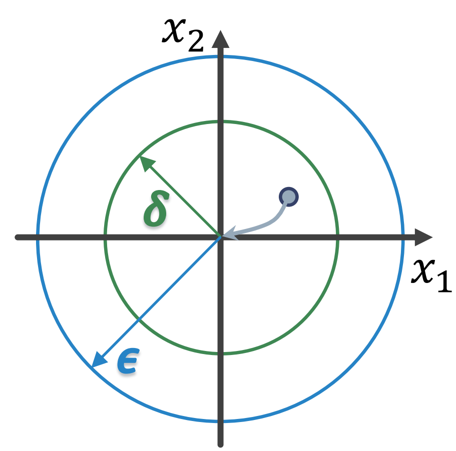
An equilibrium point $\boldsymbol{x}_e = 0$ is exponentially stable if there exists coefficient $m \geq 0$ and rate $\alpha \geq 0$ such that

$$\|\boldsymbol{x}(\boldsymbol{t})\| \leq \|\boldsymbol{x}_o\| m e^{-\alpha(t-t_0)}$$

For all $\boldsymbol{x}_o$ in some ball around $\boldsymbol{x}_e = 0$.



$x_e = 0$



$\mathbf{x}_e = [0 \quad 0]^{\mathrm{T}}$

Local vs Global

These are local definitions of stability about an equilibrium point.

- We were free to choose small δ in order to start $\boldsymbol{x}_0$ near $\boldsymbol{x}_e$.

We say an equilibrium point $\boldsymbol{x}_e$ **is globally stable** if it is stable for all initial conditions $\boldsymbol{x}_0$.

Stability of LTI Systems

Linear-Time Invariant (LTI) systems:

$$\dot{\mathbf{x}} = A\mathbf{x}$$

$$\mathbf{x} \in \mathbb{R}^n$$
$$A \in \mathbb{R}^{nxn}, \text{constant}$$

- An LTI system is **asymptotically stable** if and only if all the eigenvalues of A have **strictly negative** real parts.
- For LTI systems, asymptotic stability $\Leftrightarrow$ exponential stability.
- The system is **marginally stable** if and only if all the eigenvalues of A have **nonpositive** real parts, at least one has zero real part, *and every eigenvalue with zero real parts has its algebraic multiplicity equal to it's geometric multiplicity.*

Control of a First Order System

Problem
- State $\mathbf{x}$ and input $\mathbf{u}$
- Kinematic model $\dot{\mathbf{x}} = \mathbf{u}$ ← u is a velocity
- Want to follow trajectory $\mathbf{x}^{\mathrm{des}}(t)$

General approach
- Define error $\mathbf{e}(t) = \mathbf{x}^{\mathrm{des}}(t) - \mathbf{x}(t)$
- Want $\mathbf{e}(t)$ to converge exponentially to 0

Strategy
- Find $\mathbf{u}$ such that $\dot{\mathbf{e}} + K_p \mathbf{e} = 0$
- If $K_p > 0$ then $\mathbf{e}(t) = \exp\big(-K_p(t - t_0)\big)\,\mathbf{e}(t_0)$
- $\mathbf{u}(t) = \dot{\mathbf{x}}^{\mathrm{des}}(t) + K_p \mathbf{e}(t)$

Control of a Second Order System

Problem

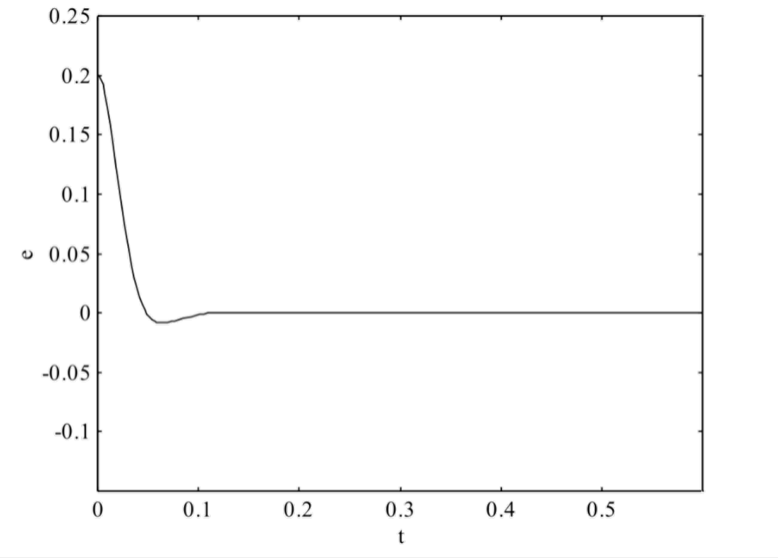
- State $\mathbf{x}$ and input $\mathbf{u}$
- Kinematic model $\ddot{\mathbf{x}} = \mathbf{u}$ ← u is an acceleration
- Want to follow trajectory $\mathbf{x}^{\text{des}}(t)$

General approach

- Define error $\mathbf{e}(t) = \mathbf{x}^{\text{des}}(t) - \mathbf{x}(t)$
- Want $\mathbf{e}(t)$ to converge exponentially to 0

Strategy

- Find $\mathbf{u}$ such that $\ddot{\mathbf{e}} + K_d\dot{\mathbf{e}} + K_p\mathbf{e} = 0$
- Pick some $K_p, K_d > 0$
- $\mathbf{u}(t) = \ddot{\mathbf{x}}^{\text{des}}(t) + K_d\dot{\mathbf{e}}(t) + K_p\mathbf{e}(t)$



Control for Trajectory Tracking

PD Control

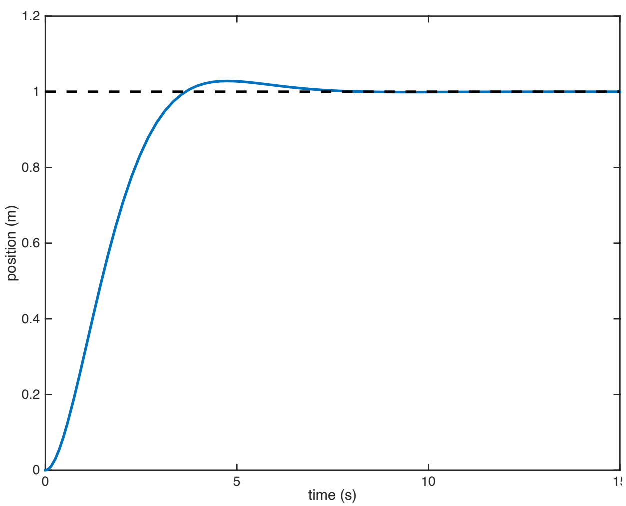
- $\mathbf{u}(t) = \ddot{\mathbf{x}}^{\mathrm{des}}(t) + K_d\dot{\mathbf{e}}(t) + K_p\mathbf{e}(t)$
- Proportional term (K_p) has a spring (capacitance) response
- Derivative term (K_d) has a dashpot (resistance) response

PID Control

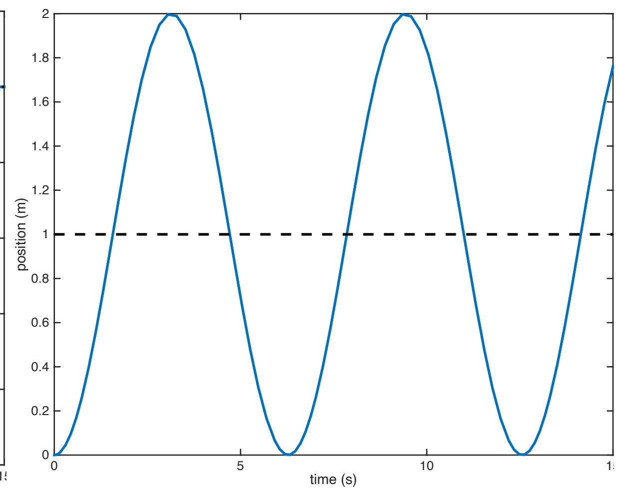
- $\mathbf{u}(t) = \ddot{\mathbf{x}}^{\mathrm{des}}(t) + K_d\dot{\mathbf{e}}(t) + K_p\mathbf{e}(t) + K_I\int_0^t \mathbf{e}(\tau)d\tau$
- Integral term (K_I) makes steady state error go to 0
  - Accounts for model error or disturbances
- PID control generates a third-order closed-loop system

Control Gains

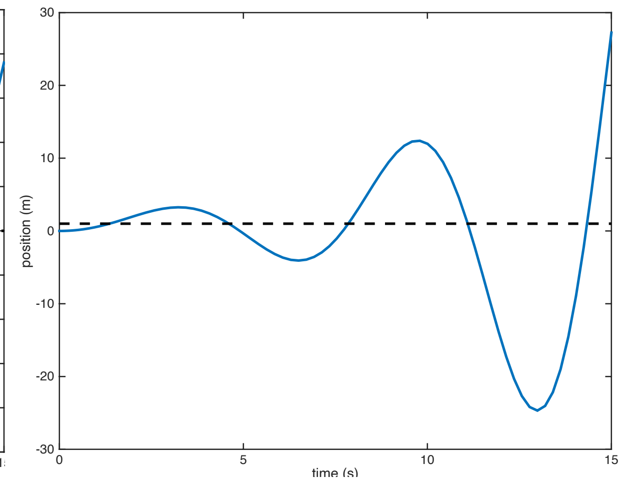
Gains change the system response



Stable



Marginally Stable



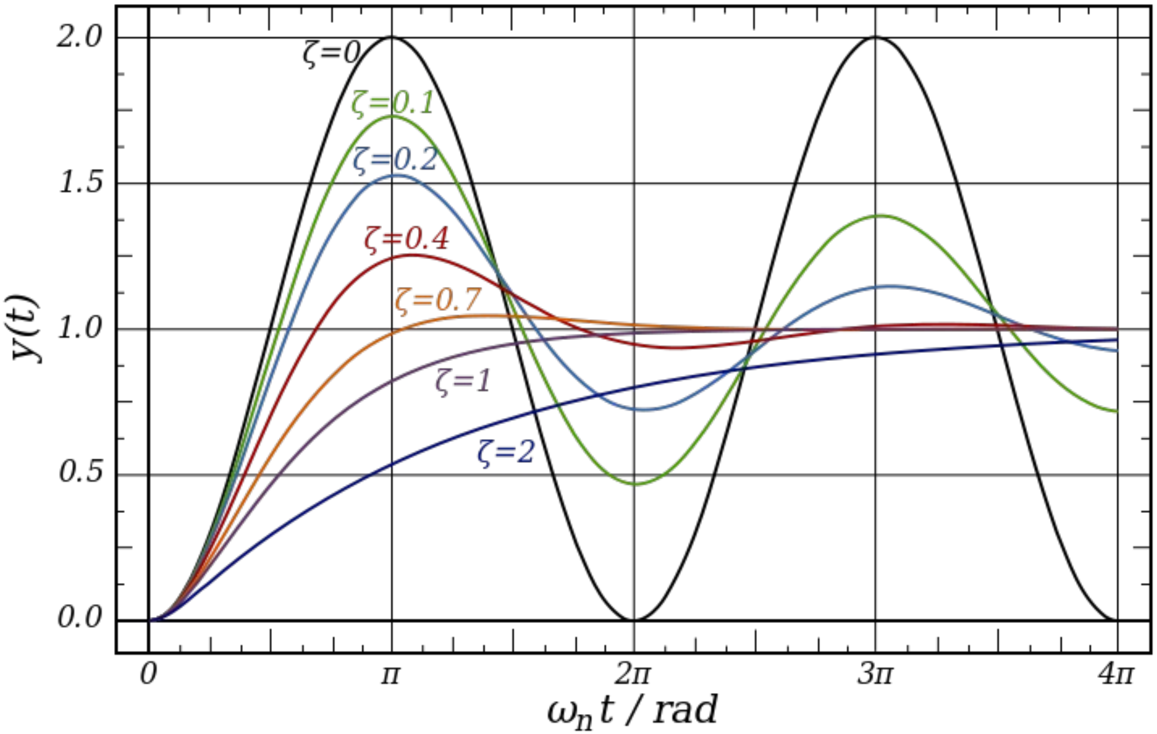
Unstable

Stereotyped 2nd Order Response

$$\ddot{e} + K_d\dot{e} + K_p e = 0$$

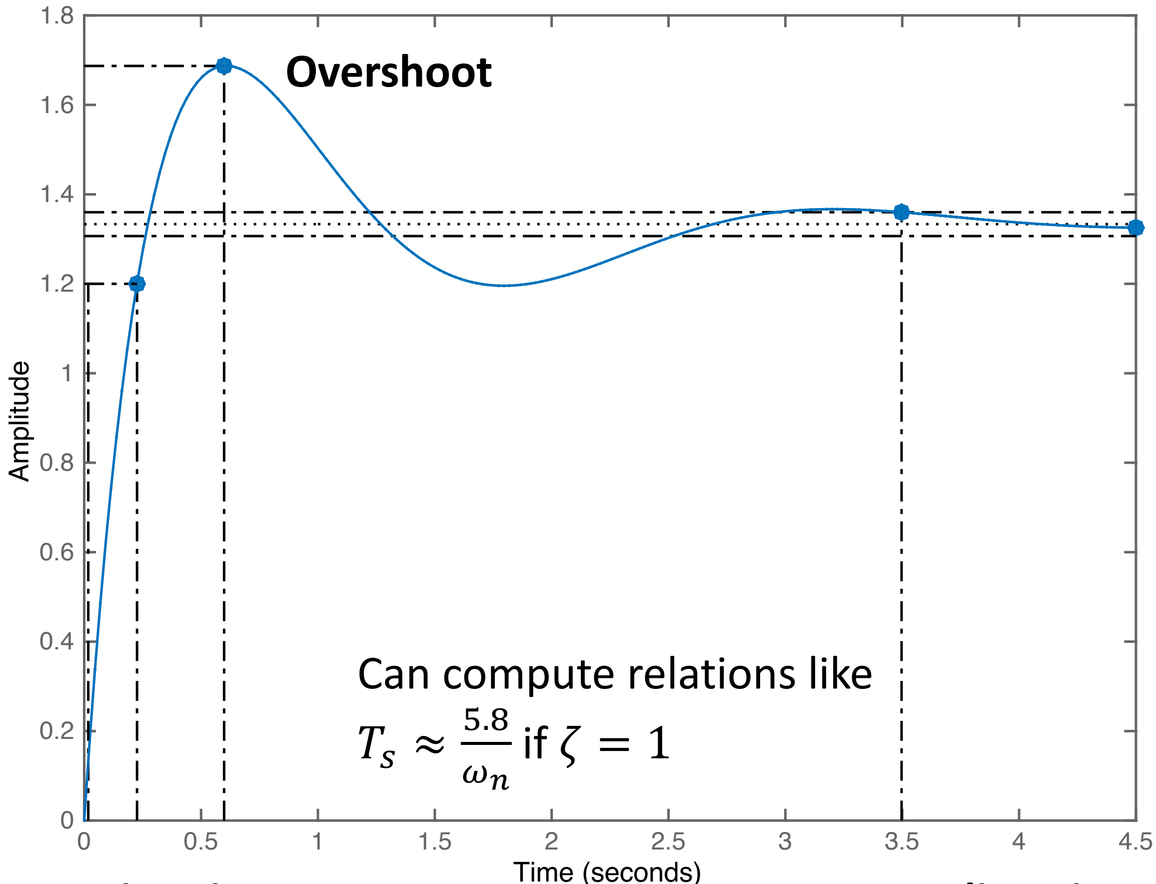
$$\ddot{e} + 2\zeta\omega_n\dot{e} + {\omega_n}^2 e = 0$$

$$\lambda = -\omega_n(\zeta \pm i\sqrt{1-\zeta^2})$$



For this simple example, can choose K_p and K_d to get a desired damping ratio.

Response to Disturbance



Overshoot

Steady-State

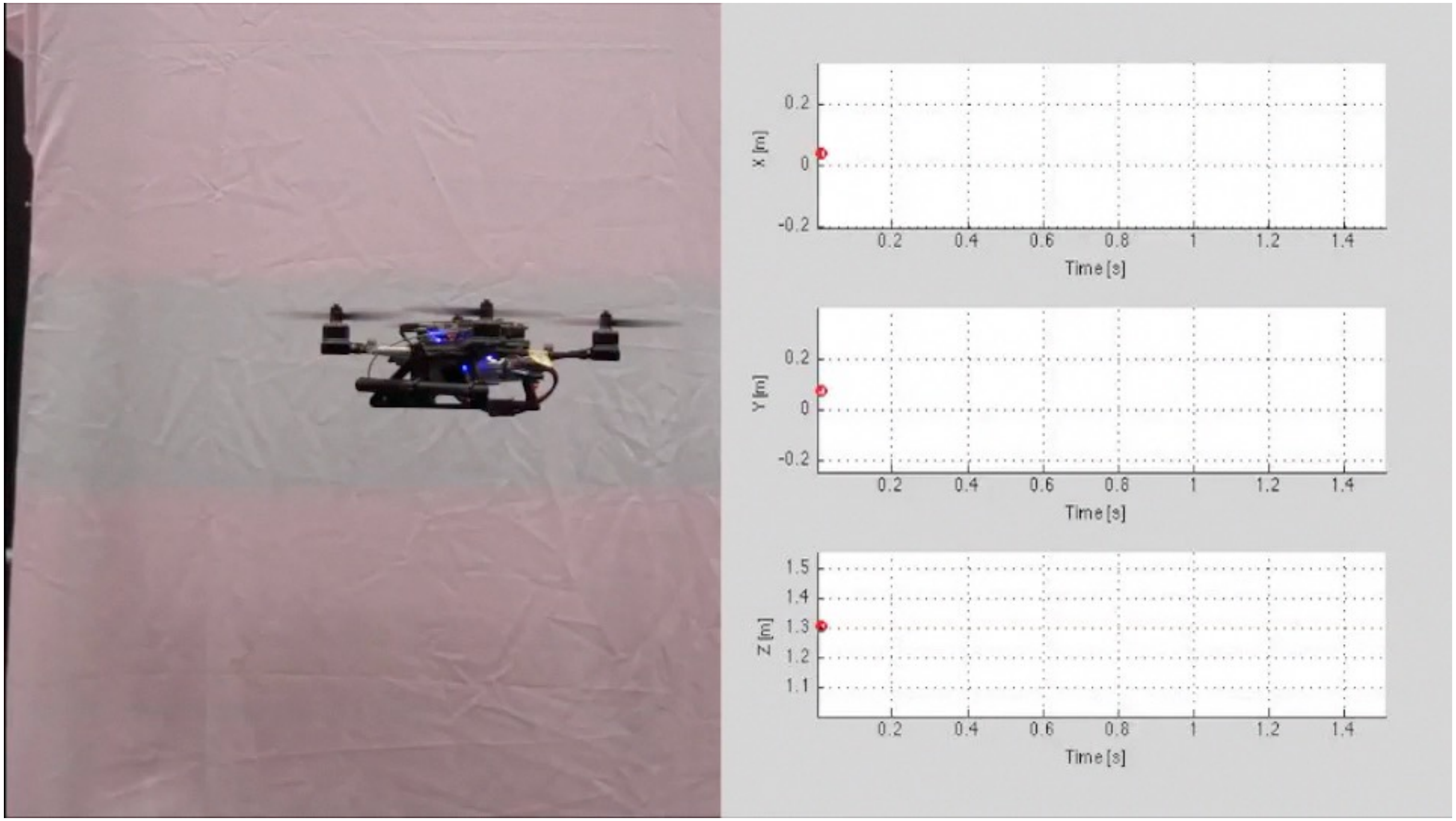
Rise Time

Settling Time

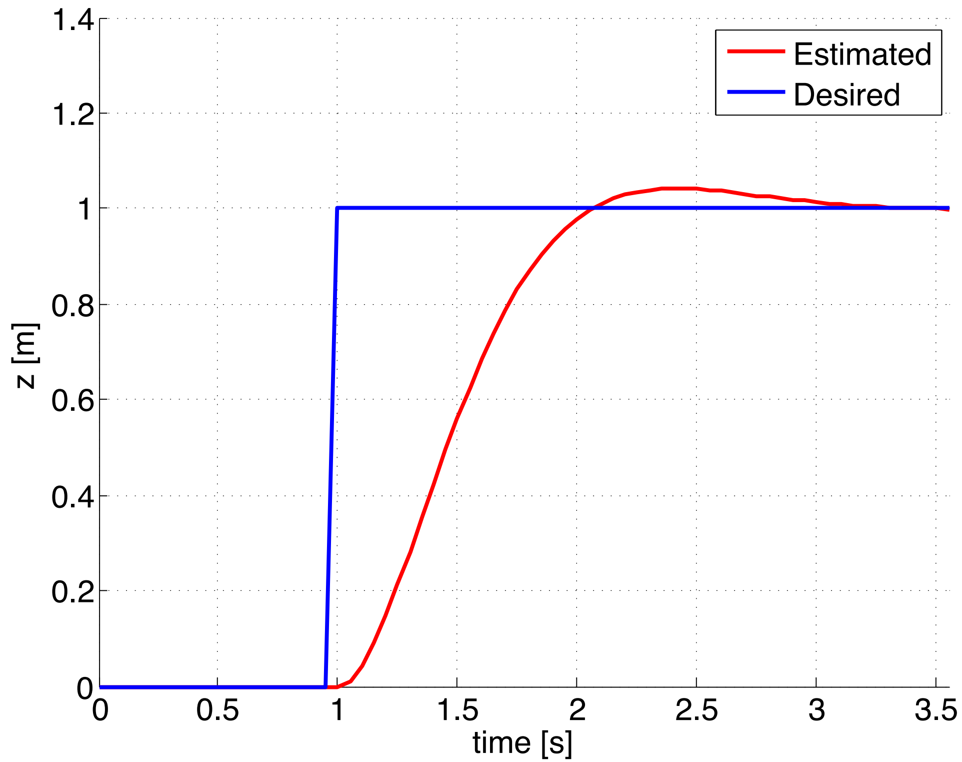
Can compute relations like

$T_s \approx \frac{5.8}{\omega_n}$ if $\zeta = 1$

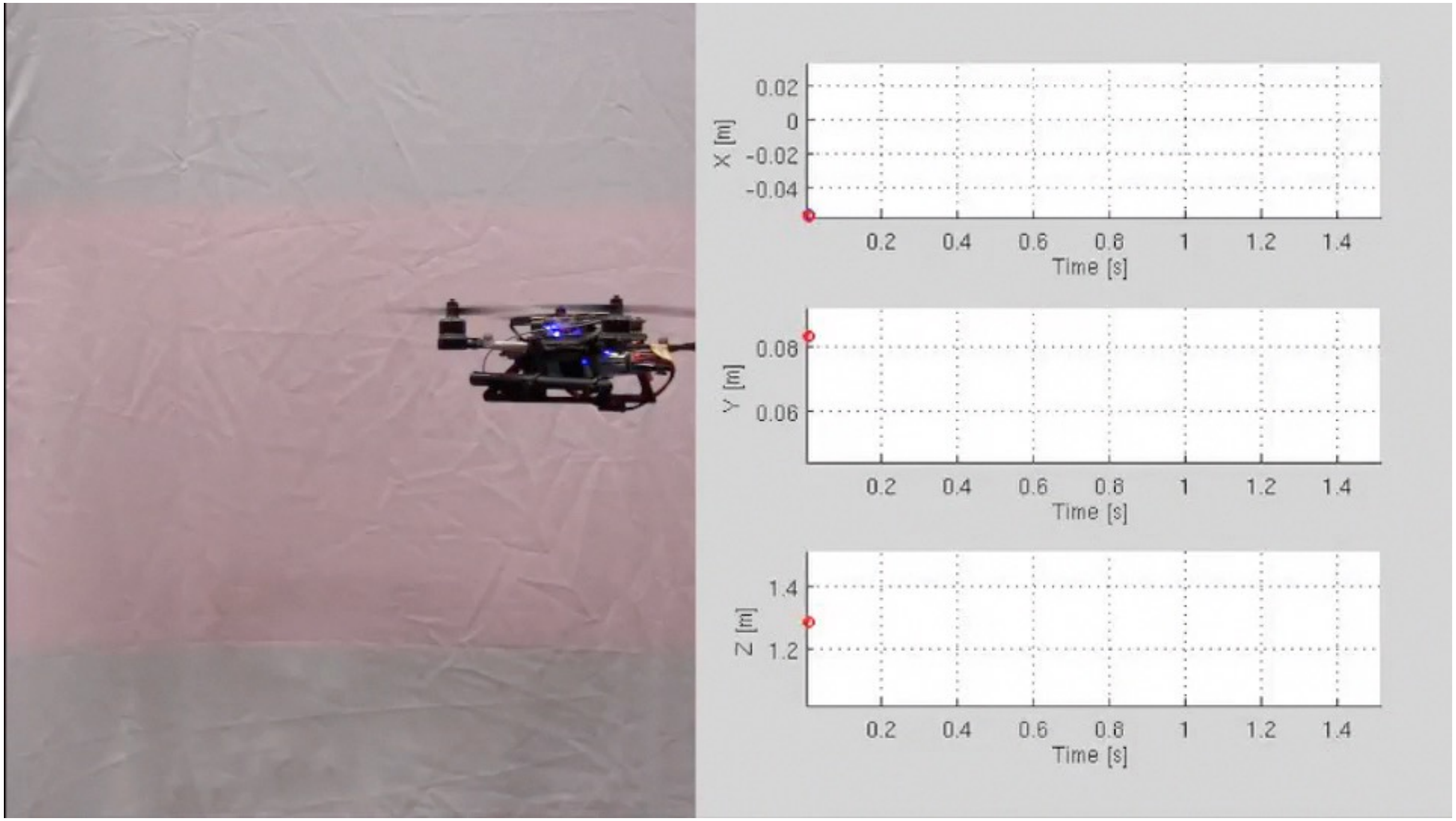
PD Position Controller



PD Controller for Z

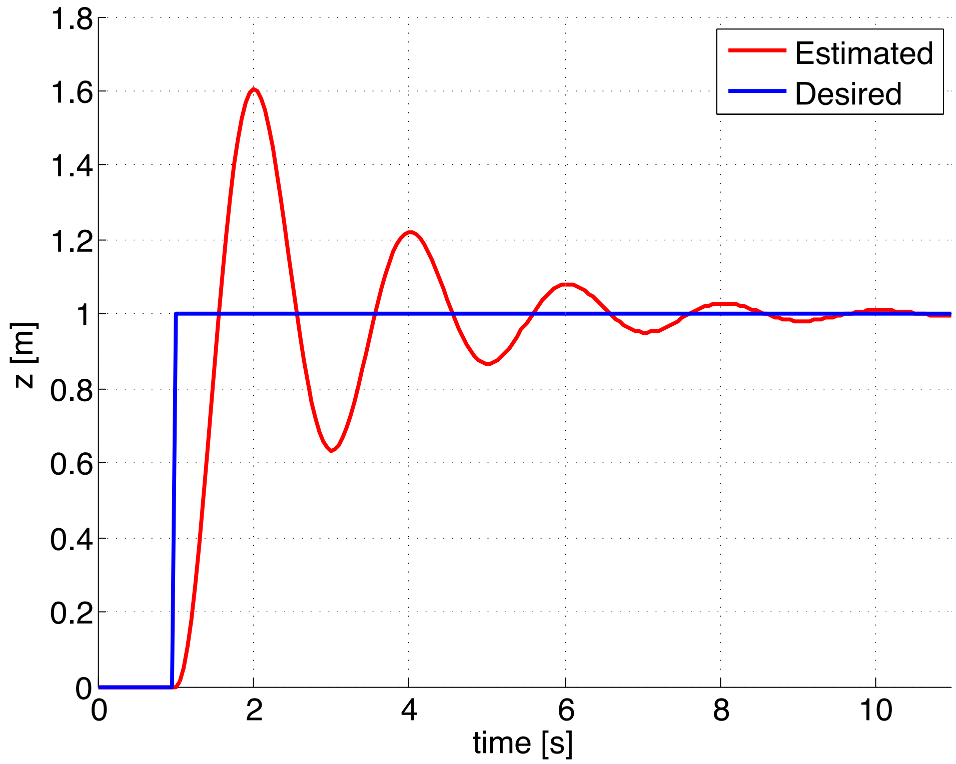


High K_p

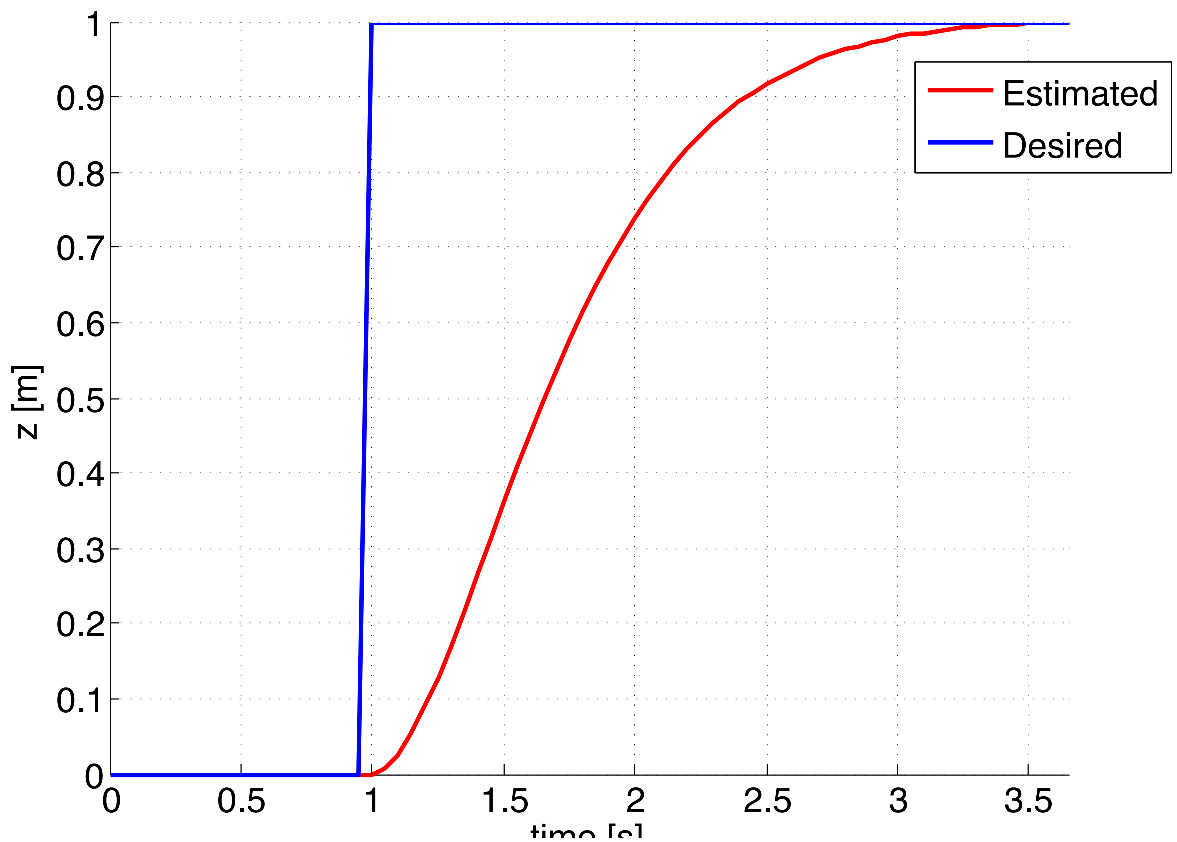


0.02
0
-0.02
-0.04
X [m]
0.2 0.4 0.6 0.8 1 1.2 1.4
Time [s]
0.08
0.06
Y [m]
0.2 0.4 0.6 0.8 1 1.2 1.4
Time [s]
1.4
1.2
Z [m]
0.2 0.4 0.6 0.8 1 1.2 1.4
Time [s]

High K_p

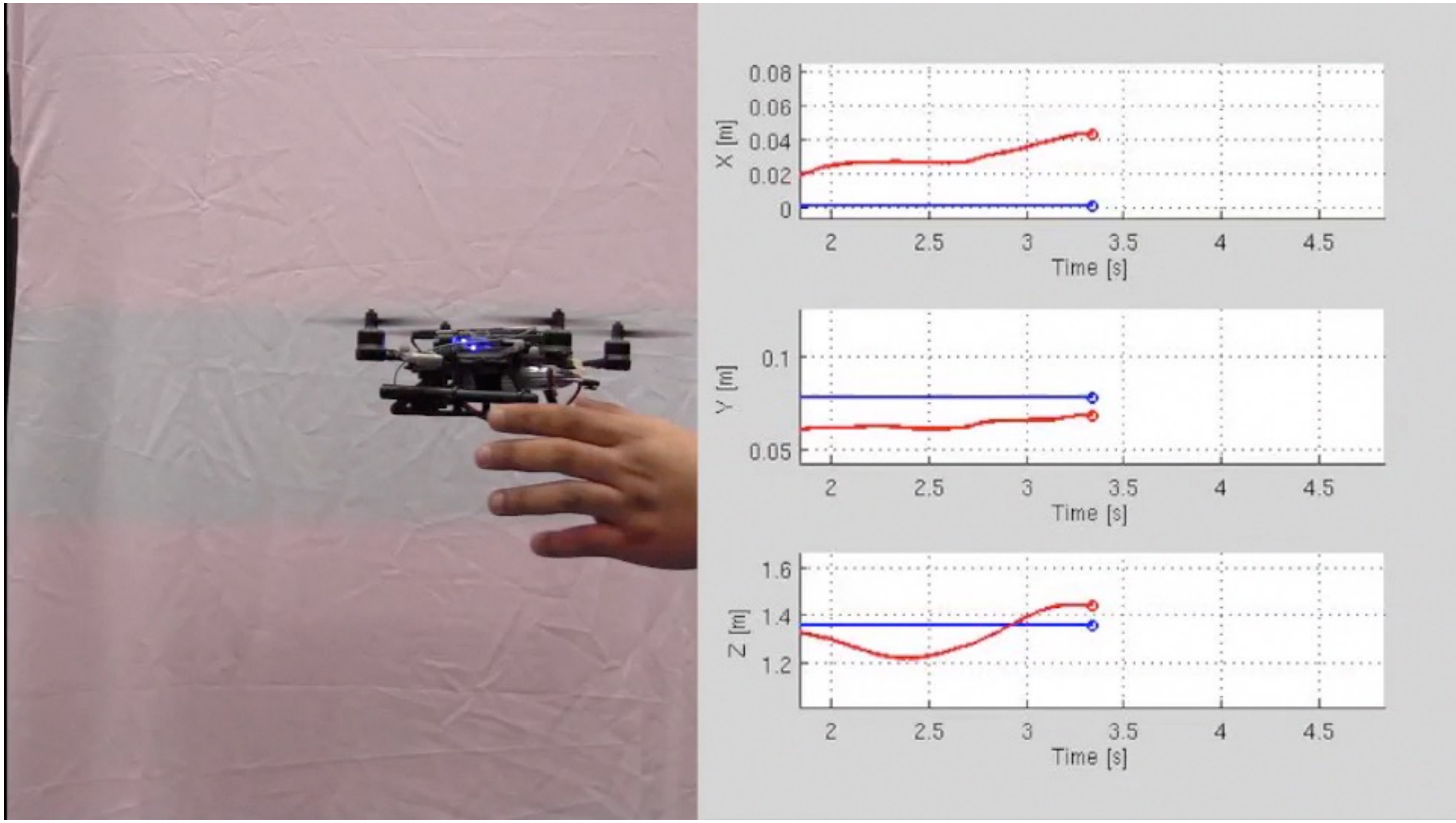


Low K_p



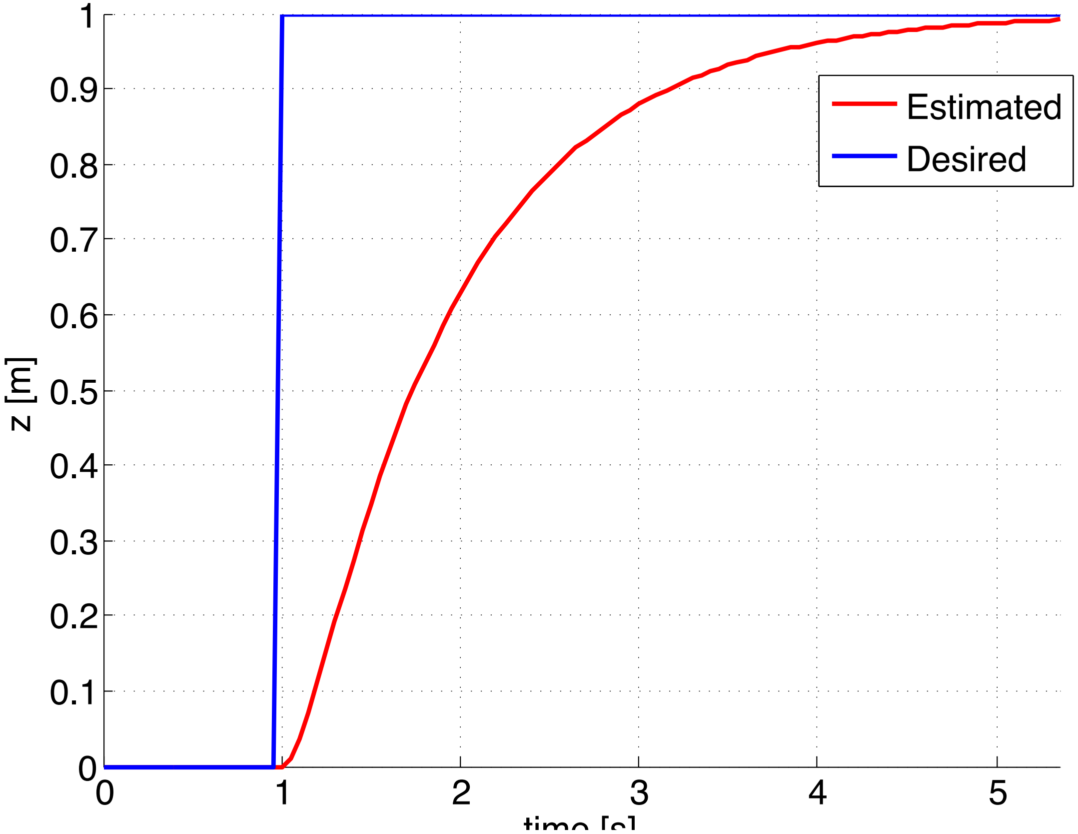
Estimated
Desired
z [m]
time [s]
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
0.5
1
1.5
2
2.5
3
3.5

High K_d



0.08
0.06
0.04
0.02
0
X [m]
2
2.5
3
3.5
4
4.5
Time [s]
0.1
0.05
Y [m]
2
2.5
3
3.5
4
4.5
Time [s]
1.6
1.4
1.2
Z [m]
2
2.5
3
3.5
4
4.5
Time [s]

High K_d



Manual Tuning

Parameter Increased	K_p	K_d	K_I
Rise Time	Decrease	-	Decrease
Overshoot	Increase	Decrease	Increase
Settling Time	-	Decrease	Increase
Steady-State Error	Decrease	-	Eliminate

- "If I increase $\boldsymbol{K_P}$, then
  Rise Time will ***Decrease,*** and
  Overshoot will ***Increase,*** and
  Steady-State Error will ***Decrease."***
- These are only general guidelines for "typical systems."

Ziegler-Nichols Method

Heuristic method for PID gain tuning

1. Set $K_d = K_I = 0$
2. Increase K_p until ultimate gain K_u where system starts to oscillate
3. Find oscillation period T_u at K_u
4. Set gains according to:

Controller	K_p	K_d	K_I
P	$0.5K_u$	--	--
PD	$0.8K_u$	$K_pT_u/8$	--
PID	$0.6K_u$	$K_pT_u/8$	$2K_p/T_u$

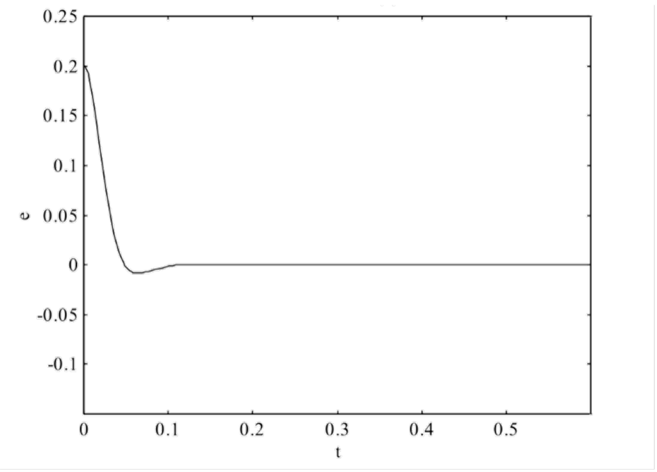
Model-Based Control

PD and PID control laws applied to real systems

- $m\ddot{\mathbf{x}}(t) + b\dot{\mathbf{x}}(t) + k\mathbf{x}(t) = \boldsymbol{u}(t)$ ← u is a force! also, system dynamics!
- Performance will depend on the system dynamics
- Need to tune gains to maximize performance

Model-based control law

- $\mathbf{u}(t) = m\left(\ddot{\mathbf{x}}^{\text{des}}(t) + K_d\dot{\mathbf{e}}(t) + K_p\mathbf{e}(t)\right) + b\dot{\mathbf{x}}(t) + k\mathbf{x}(t)$
- Servo-based component
  - Use PD (or PID) feedback to drive error to 0
  - Independent of the model
- Model-based component
  - Cancels system dynamics
  - Specific to the model



Model-Based Control

Advantages

- Decomposes control law model-dependent and model-independent part
- Model-independent gains will work for any system

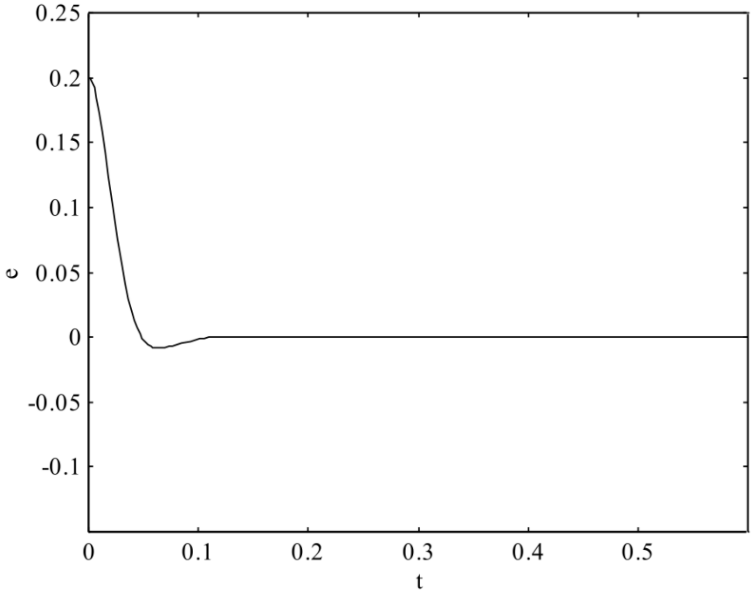
Disadvantages

- If model parameters have errors then error will not go to 0
- Original system
  - $m\ddot{\mathbf{x}}(t) + b\dot{\mathbf{x}}(t) + k\mathbf{x}(t) = \boldsymbol{u}(t)$
- Our control law
  - $\mathbf{u}(t) = \hat{m}\left(\ddot{\mathbf{x}}^{\mathrm{des}}(t) + K_d\dot{\mathbf{e}}(t) + K_p\mathbf{e}(t)\right) + \hat{b}\dot{\mathbf{x}}(t) + \hat{k}\mathbf{x}(t)$

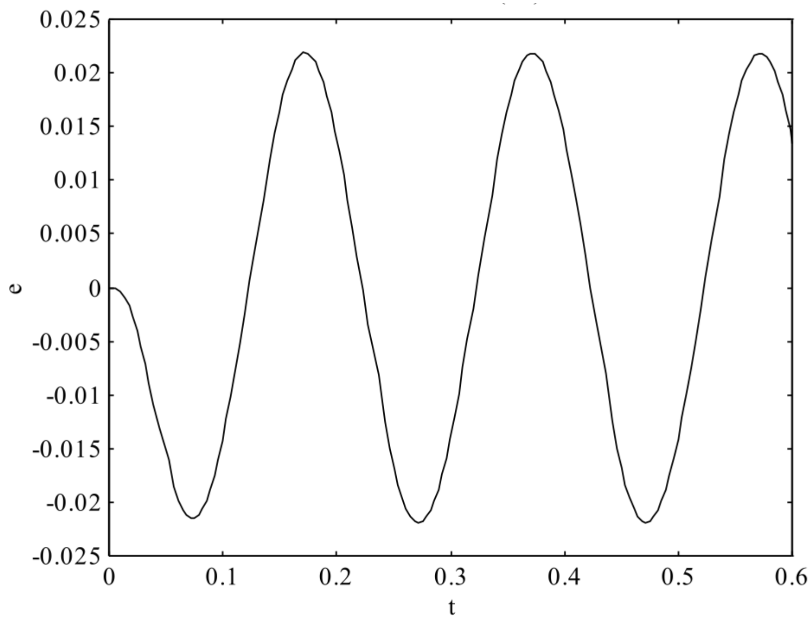
    Estimates
- Substitute to find total system dynamics
  - $\ddot{\mathbf{e}} + K_d\dot{\mathbf{e}} + K_p\mathbf{e} = \left(\frac{m}{\hat{m}} - 1\right)\ddot{\mathbf{x}} + \frac{b-\hat{b}}{\hat{m}}\dot{\mathbf{x}} + \frac{k-\hat{k}}{\hat{m}}\mathbf{x}$
- Right-hand side drives error away from 0!

Model-Based Control

- $\ddot{\mathbf{e}} + K_d\dot{\mathbf{e}} + K_p\mathbf{e} = \left(\frac{m}{\hat{m}} - 1\right)\ddot{\mathbf{x}} + \frac{b-\hat{b}}{\hat{m}}\dot{\mathbf{x}} + \frac{k-\hat{k}}{\hat{m}}\mathbf{x}$



Perfect model

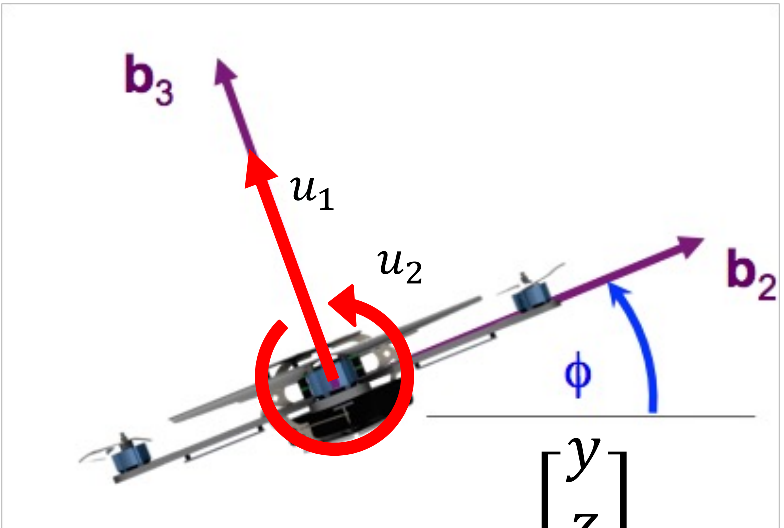


Imperfect model – 10% errors

If right-hand side is bounded then we can prove $\mathbf{e}(t)$ also bounded

Quadrotor Control

Planar Quadrotor Model



$$\begin{bmatrix} \ddot{y} \\ \ddot{z} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 \\ -g \\ 0 \end{bmatrix} + \begin{bmatrix} -\frac{1}{m}\sin\phi & 0 \\ \frac{1}{m}\cos\phi & 0 \\ 0 & \frac{1}{I_{xx}} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} y \\ z \\ \phi \\ \dot{y} \\ \dot{z} \\ \dot{\phi} \end{bmatrix}$$

$$\dot{\mathbf{x}} = \begin{bmatrix} x_4 \\ x_5 \\ x_6 \\ 0 \\ -g \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -m^{-1}\sin x_3 & 0 \\ m^{-1}\cos x_3 & 0 \\ 0 & I_{xx}^{-1} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Affine Nonlinear System

State $\mathbf{x}$ and input $\mathbf{u}$

State equations $\dot{\mathbf{x}} = f(\mathbf{x}) + g(\mathbf{x})\mathbf{u}$

- $\dot{x} = \begin{bmatrix} x_4 \\ x_5 \\ x_6 \\ 0 \\ -g \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -m^{-1}\sin x_3 & 0 \\ m^{-1}\cos x_3 & 0 \\ 0 & I_{xx}^{-1} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$

- Nonlinear in the state
- Affine in the control input

Linearized Dynamics

Nonlinear dynamics

$$\ddot{y} = -\frac{u_1}{m}\sin(\phi)$$

$$\ddot{z} = -g + \frac{u_1}{m}\cos(\phi)$$

$$\ddot{\phi} = \frac{u_2}{I_{xx}}$$

Linearized model

$$\ddot{y} = -g\phi$$

$$\ddot{z} = \frac{u_1}{m}$$

$$\ddot{\phi} = \frac{u_2}{I_{xx}}$$

Equilibrium configuration

$$\mathbf{q}_e = \begin{bmatrix} y_0 \\ z_0 \\ 0 \end{bmatrix}, \mathbf{x}_e = \begin{bmatrix} \mathbf{q}_e \\ \mathbf{0} \end{bmatrix}$$

Equilibrium hover condition

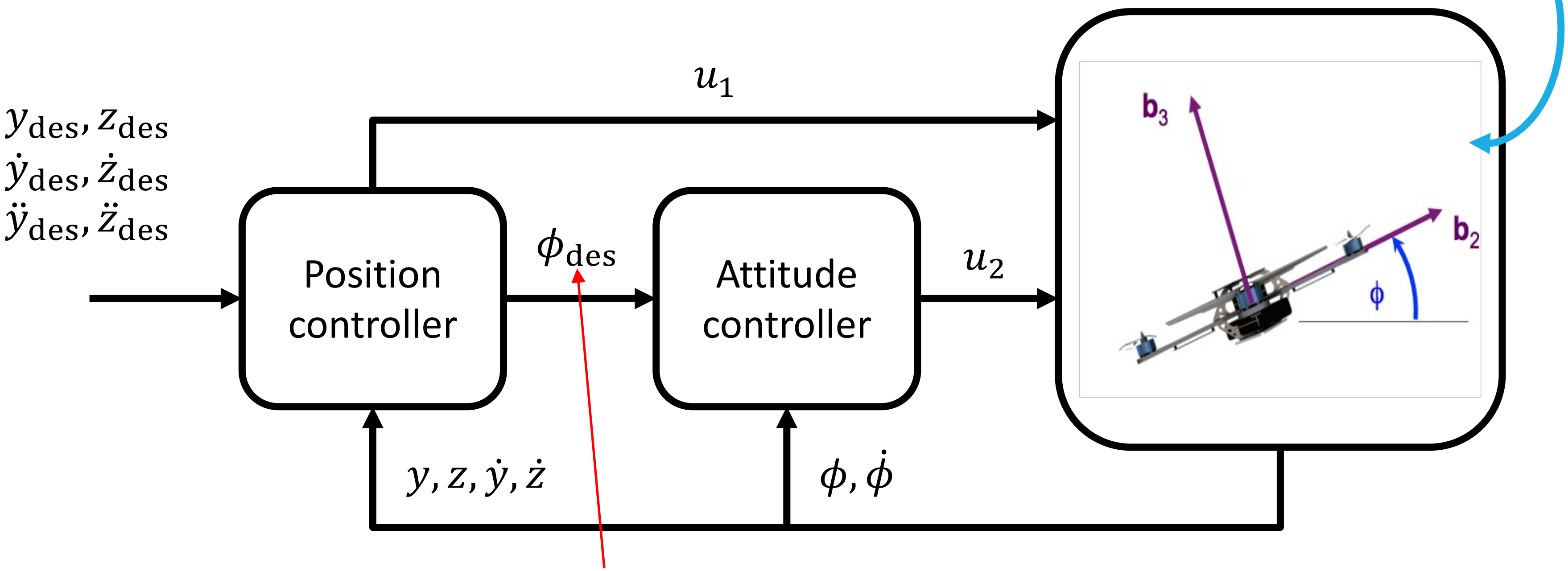
y_0, z_0

$\phi_0 = 0$

$u_{1,0} = mg, u_{2,0} = 0$

$$\ddot{y} = -g\phi$$
$$\ddot{z} = -g + \frac{u_1}{m}$$
$$\ddot{\phi} = \frac{u_2}{I_{xx}}$$

Nested Control Structure



Specified by the position controller, **not** the user

Works when inner (attitude) control loop runs much faster (10x) than the outer (position) control loop

Control Equations

Recall for a second order system $\ddot{\mathbf{e}} + K_d\dot{\mathbf{e}} + K_p\mathbf{e} = 0$

For any configuration variable q we have

commanded

actual (feedback)

$$(\ddot{q}_{\mathrm{des}} - \ddot{q}) + K_{d,q}(\dot{q}_{\mathrm{des}} - \dot{q}) + K_{p,q}(q_{\mathrm{des}} - q) = 0$$

specified

Control Equations

Lateral dynamics

- $\ddot{y} = -g\phi$
- $\ddot{\phi} = \frac{u_2}{I_{xx}}$

Desired attitude

- $\phi_{\text{des}} = -\frac{\ddot{y}_c}{g}$
- $\dot{\phi}_{\text{des}} = 0$
- $\ddot{\phi}_{\text{des}} = 0$

Attitude controller

- $u_2 = I_{xx}\ddot{\phi}_c$

Vertical dynamics

- $\ddot{z} = \frac{u_1}{m}$

Z-position controller

- $u_1 = m(\ddot{z}_c)$

Control Equations

Control equations

$$u_1 = m\left(\ddot{z}_{\text{des}} + k_{d,z}(\dot{z}_{\text{des}} - \dot{z}) + k_{p,z}(z_{\text{des}} - z)\right)$$

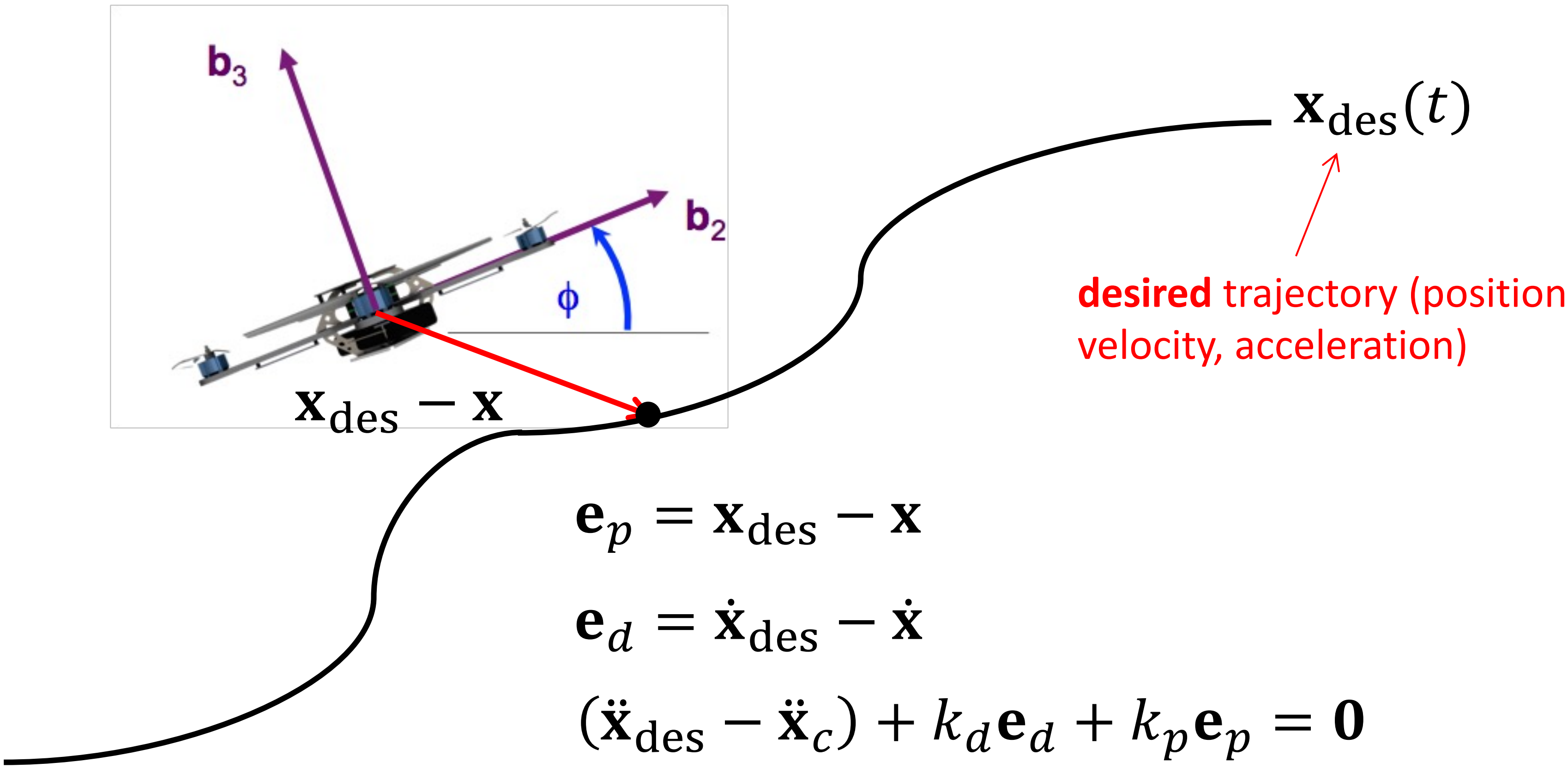
$$u_2 = I_{xx}\left(\ddot{\phi}_{\text{des}} + k_{d,\phi}\left(\dot{\phi}_{\text{des}} - \dot{\phi}\right) + k_{p,\phi}(\phi_{\text{des}} - \phi)\right)$$

$$\phi_{\text{des}} = -\frac{1}{g}\left(\ddot{y}_{\text{des}} + k_{d,y}(\dot{y}_{\text{des}} - \dot{y}) + k_{p,y}(y_{\text{des}} - y)\right)$$

- Three sets of PD gains.
- Systematically tune using step responses.
    - Thrust
    - Roll
    - Position (depends on roll being well-tuned)

Trajectory Tracking (in time)

Follow trajectory exactly



$$\mathbf{e}_p = \mathbf{x}_{\text{des}} - \mathbf{x}$$

$$\mathbf{e}_d = \dot{\mathbf{x}}_{\text{des}} - \dot{\mathbf{x}}$$

$$(\ddot{\mathbf{x}}_{\text{des}} - \ddot{\mathbf{x}}_c) + k_d \mathbf{e}_d + k_p \mathbf{e}_p = \mathbf{0}$$

Project

Make plots in your sandbox!

Run tests locally.

Gain tuning: Finding 12 magic numbers by trial and error won’t work.

How to judge controller quality.

Order of tuning the cascaded controller.

Choose reasonable trajectories.

Practical constraints: actuator limits.