MEAM 620

Vijay Kumar and James Paulos

NONLINEAR CONTROL

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

Model-Based Control

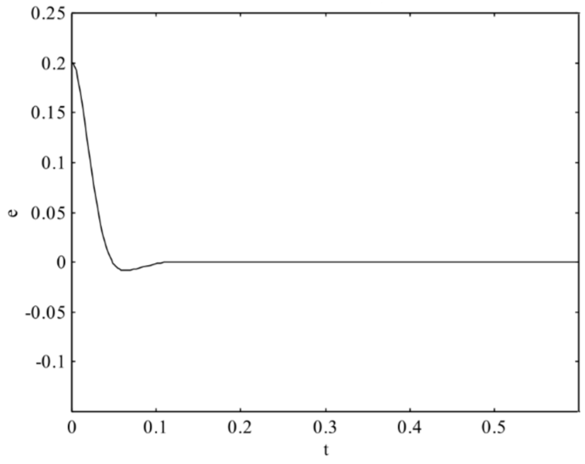
PD and PID control laws applied to real systems

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

u is a force!
also, system dynamics!

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

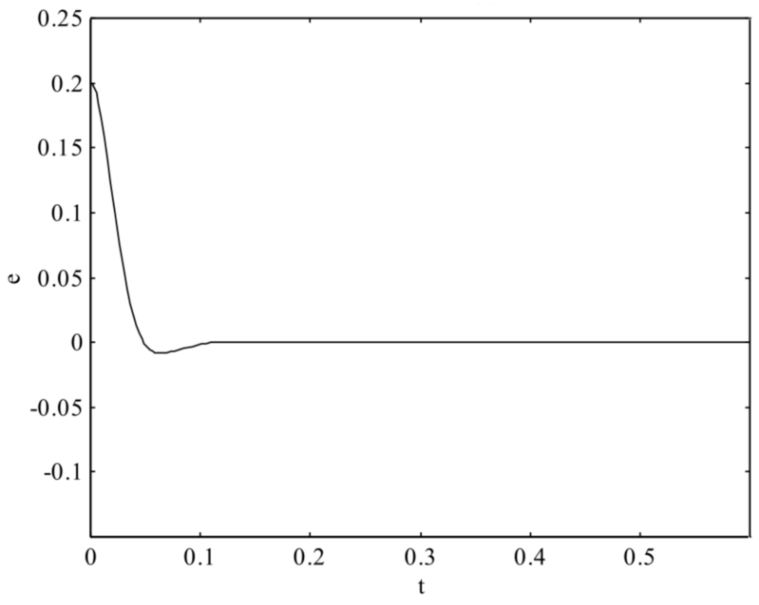
Disadvantages

- 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
  - $\mathbf{u}(t) = \hat{m}\left(\ddot{\mathbf{x}}^{\text{des}}(t) - K_d\dot{\mathbf{e}}(t) - K_p\mathbf{e}(t)\right) + \hat{b}\dot{\mathbf{x}}(t) + \hat{k}\mathbf{x}(t)$

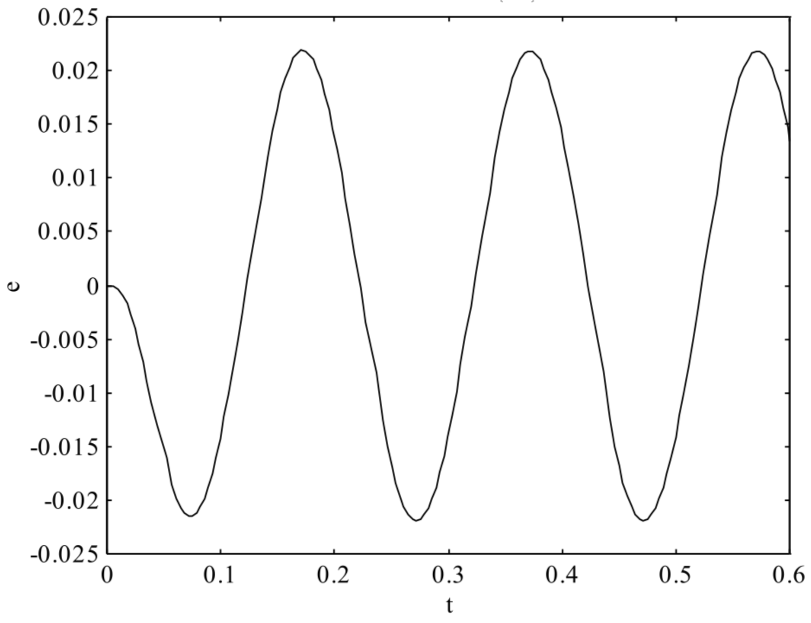
    Estimates
- Substitute to find total system dynamics
  - $\ddot{\mathbf{e}} + K_d\dot{\mathbf{e}} + K_p\mathbf{e} = \left(1 - \frac{m}{\hat{m}}\right)\ddot{\mathbf{x}} + \frac{\hat{b}-b}{\hat{m}}\dot{\mathbf{x}} + \frac{\hat{k}-k}{\hat{m}}\mathbf{x}$
- Right-hand side drives error away from 0!

Model-Based Control

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



Perfect model



Imperfect model – 10% errors

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

Fully Actuated vs Underactuated

A control system with coordinates $\boldsymbol{q}$ and inputs $\boldsymbol{u}$ is **fully actuated** if it can achieve any instantaneous acceleration in $\boldsymbol{q}$.

A necessary condition is for the number of control inputs to be at least as great as the number of degrees of freedom.

Reasons for Underactuated Systems

- insufficient number of inputs
- structure of dynamics
- actuator limits

For "control-affine" systems, simple necessary and sufficient conditions for being fully actuated.

$$\ddot{\boldsymbol{q}} = f(\boldsymbol{q}, \dot{\boldsymbol{q}}) + g(\boldsymbol{q}, \dot{\boldsymbol{q}})\mathbf{u}$$

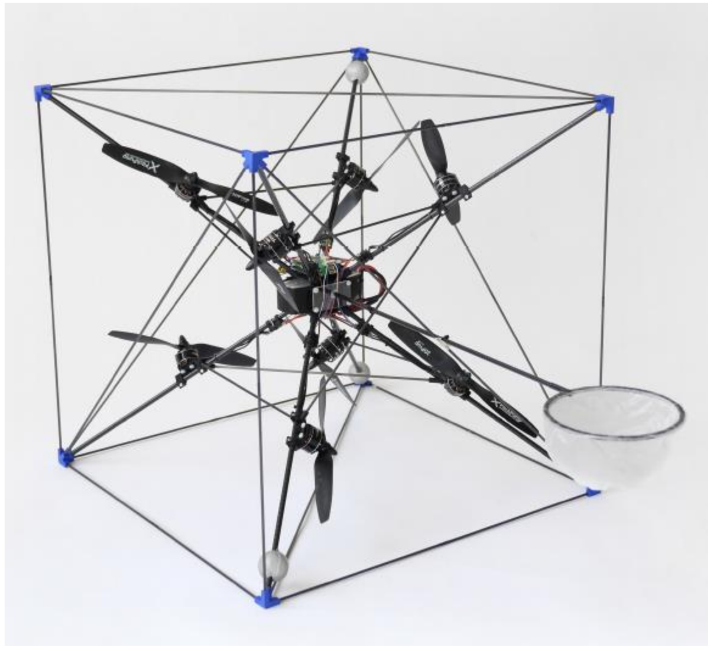
require $\operatorname{rank} g(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \dim \boldsymbol{q}$

A fully actuated multirotor?

This?

DJI

This?



Brescianini 2018

$$\ddot{\boldsymbol{q}} = f(\boldsymbol{q}, \dot{\boldsymbol{q}}) + g(\boldsymbol{q}, \dot{\boldsymbol{q}})\mathbf{u}$$

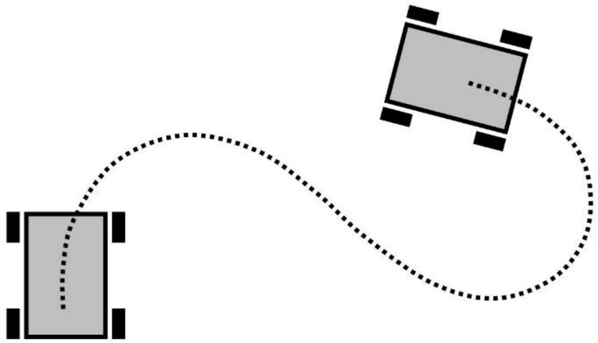
Brescianini 2016, “Design, modeling and control of an omni-directional aerial vehicle”

Holonomic and Nonholonomic

Given a dynamical system with coordinates $\boldsymbol{q}$,

- Holonomic constraints are constraints on the configuration $\boldsymbol{q}$.
- Nonholonomic constraints include constraints on the velocities $\dot{\boldsymbol{q}}$ which can not be integrated into holonomic constraints.

A car can go to any configuration $\boldsymbol{q} = (x, y, \phi)$, but it can not drive sideways. The constraint is on the *velocity*, not the *configuration*.



Nonholonomic constraints are another source of underactuated control systems.

Graphical Methods

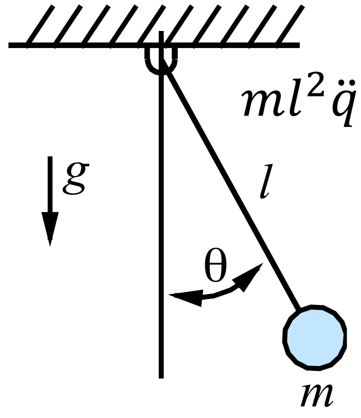
Graphical methods give a **qualitative** description of the behavior of state space systems.

- Equilibria
- Stability
- Basins of attraction

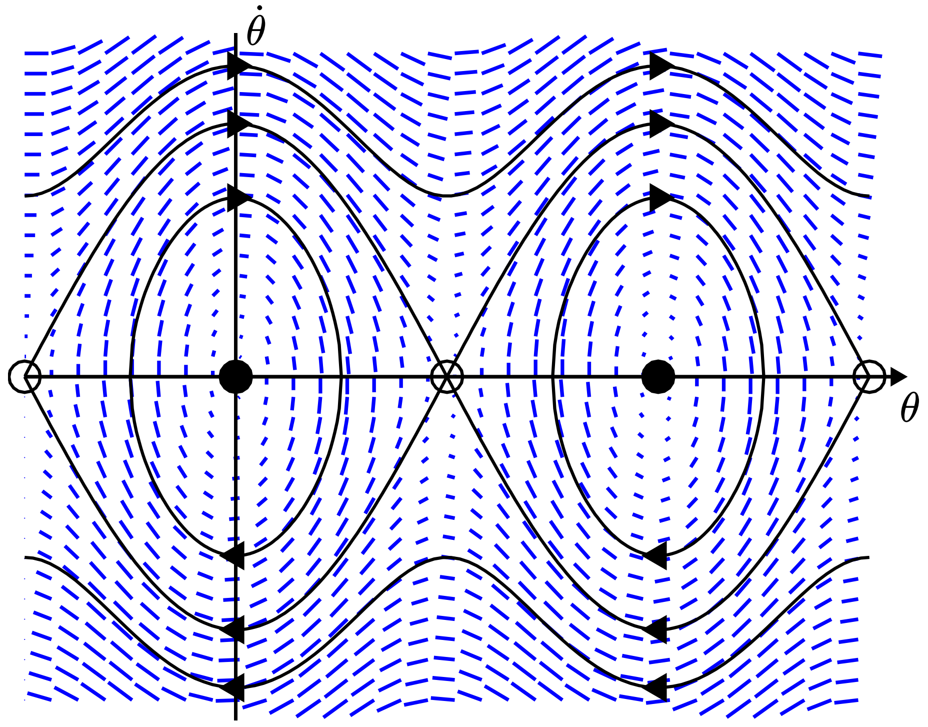
In one dimension, phase portraits can be easily sketched by hand.

Phase Portraits in 2D

In 2D, the *phase portrait* of the system $\dot{x} = f(x)$ is generated by plotting the vector field $f(x)$ over the domain of x.



$$ml^2\ddot{q} + mgl\sin q = 0$$

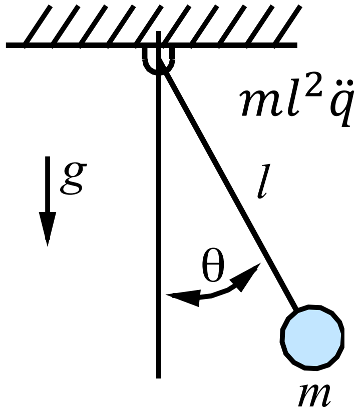


$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} q \\ \dot{q} \end{bmatrix}$$

$$\dot{x} = \begin{bmatrix} x_2 \\ -\frac{g}{l}\sin x_1 \end{bmatrix}$$

Phase Portraits in 2D

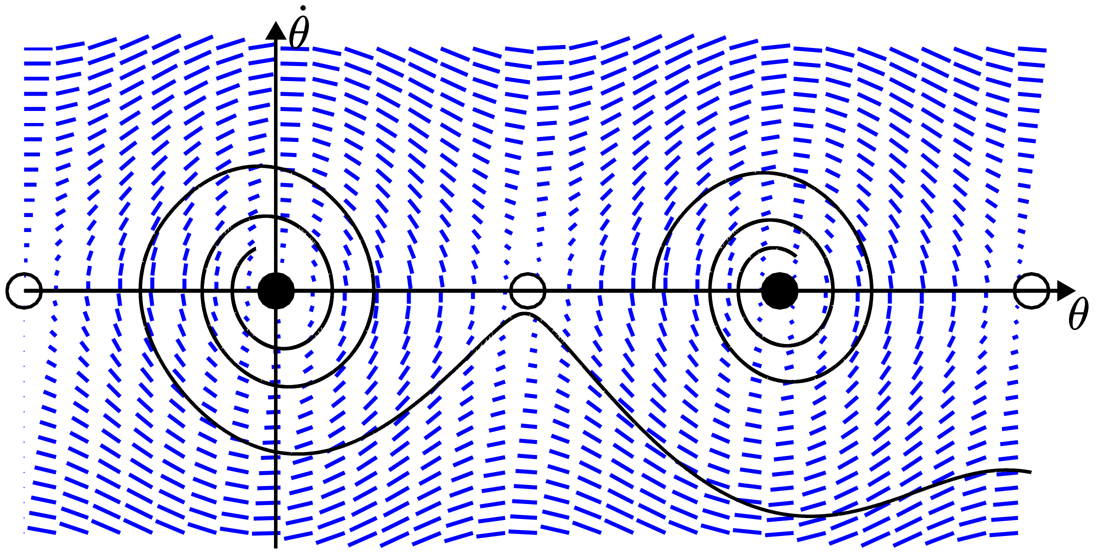
Now add damping.



$$ml^2\ddot{q} + mgl\sin q + b\dot{q} = 0$$

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} q \\ \dot{q} \end{bmatrix}$$

$$\dot{x} = \begin{bmatrix} x_2 \\ -\frac{g}{l}\sin x_1 - \frac{b}{ml^2}x_2 \end{bmatrix}$$



Lyapunov Stability Theorem

For a system

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) \qquad \boldsymbol{x} \in \mathbb{R}^n, \mathrm{f} : \mathbb{R}^n \to \mathbb{R}^n$$

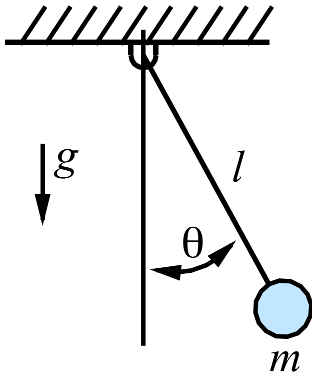
The equilibrium point $\boldsymbol{x} = 0$ is stable in $D \subset \mathbb{R}^n$ iff there exists a smooth function $V : D \subset \mathbb{R}^n \to \mathbb{R}$ such that

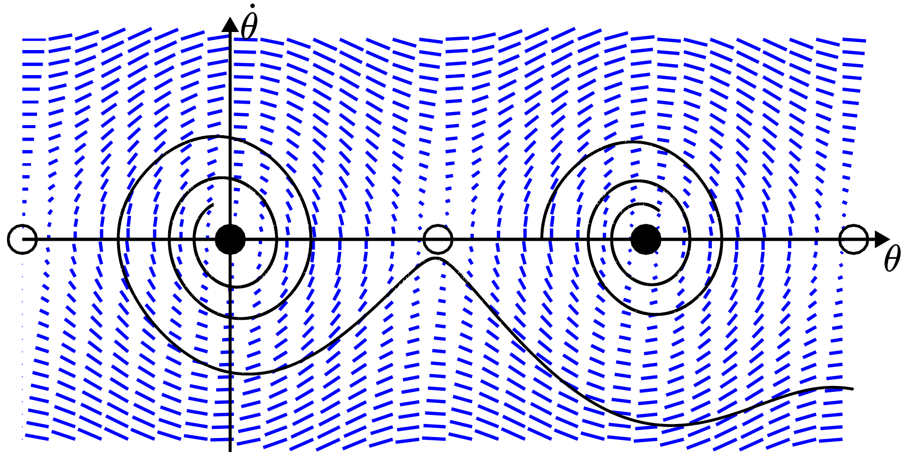
$$V(0) = 0$$

$$V > 0 \quad \forall x \in D - \{0\}$$

$$\dot{V} \leq 0 \quad \forall x \in D$$

(And if $\dot{V} < 0, \forall x \in D - \{0\}$ we have asymptotic stability.)





Lie Derivatives

system $\dot{x} = f(x)$

function $V(x)$

The Lie derivative of a function $V(x)$ along a vector field f describes how the function changes along solutions of the differential equation.

$$\frac{d}{dt}V\big(x(t)\big) = \mathcal{L}_f V(x(t))$$

$$\mathcal{L}_f V(x) = \frac{dV}{dx}(x) \cdot f(x)$$

Using this notation, Lyapunov's stability theorem requires

$$\mathcal{L}_f V(x) < 0$$

Example: Damped Pendulum

For the damped pendulum, let $V(x)$ be the total energy.

$$V(\boldsymbol{x}) = \frac{1}{2} m l^2 x_2^2 - mgl\cos x_1$$

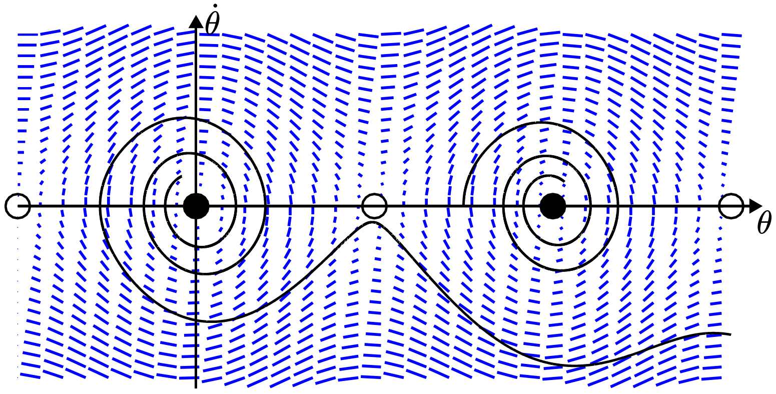
$$\dot{\boldsymbol{x}} = f(\boldsymbol{x})$$

$$f(\boldsymbol{x}) = \begin{bmatrix} x_2 \\ -\frac{g}{l}\sin x_1 - \frac{b}{ml^2} x_2 \end{bmatrix}$$

$$\mathcal{L}_f V = \frac{dV}{dx} \cdot f(x) = [mgl\sin x_1 \quad ml^2 x_2] \begin{bmatrix} x_2 \\ -\frac{g}{l}\sin x_1 - \frac{b}{ml^2} x_2 \end{bmatrix}$$

$$\mathcal{L}_f V = (x_2 mgl\sin x_1) + (-x_2 mgl\sin x_1 - bx_2^2)$$

$$\mathcal{L}_f V = -bx_2^2$$



Sadly, need to find a slightly more clever Lyapunov function to prove *asymptotic* stability.

Input-Output Linearization

Also known as partial feedback linearization.

Basic idea:

- Come up with a transformation to turn the nonlinear system into an equivalent linear system

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

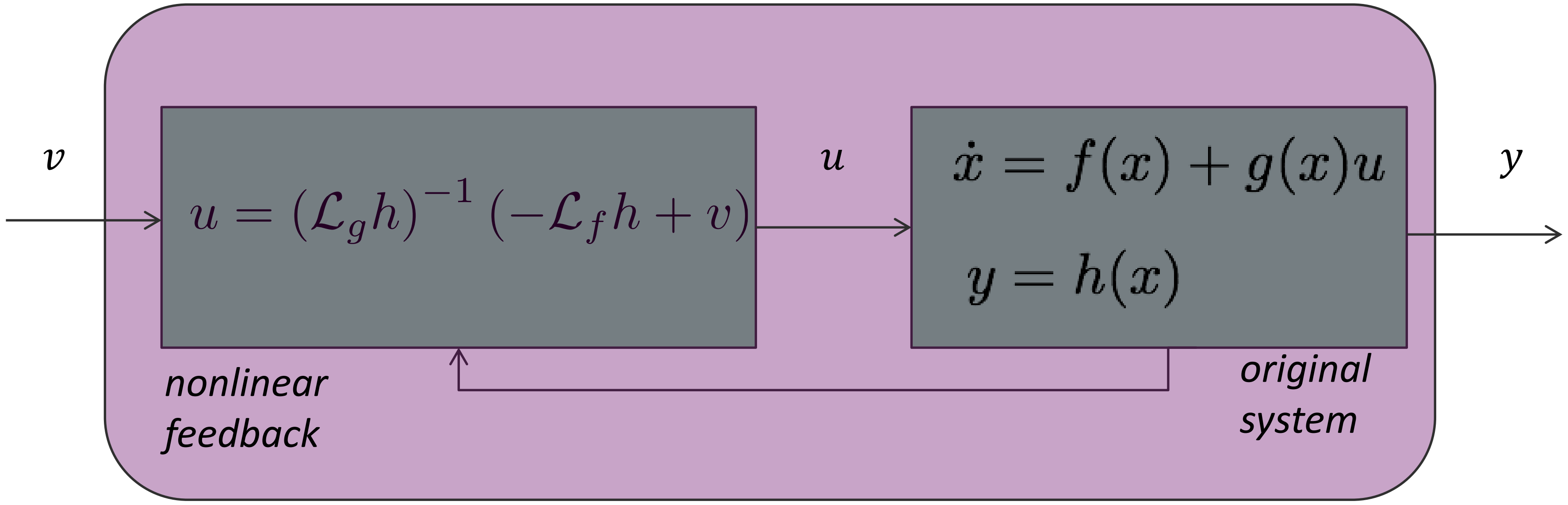
Output: $\mathbf{y} = h(\mathbf{x})$

Goal: Design a control input $\mathbf{u} = \alpha(\mathbf{x}) + \beta(\mathbf{x})\mathbf{v}$

such that $\dot{\mathbf{y}} = \mathbf{v}$

Then use the new virtual input $\boldsymbol{v}$ to control $\boldsymbol{y}$.

Input-Output Linearization



new system $$\dot{y} = v$$

Nonlinear feedback transforms the original nonlinear system to a new linear system

Linearization is exact (distinct from linear approximations to nonlinear systems)

Next step: Recipe for constructing $\mathbf{u} = \alpha(\mathbf{x}) + \beta(\mathbf{x})\mathbf{v}$

Input-Output Linearization

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

Output $y = h(x)$

Rate of change of output

$$\dot{y} = \mathcal{L}_f h + (\mathcal{L}_g h)\, u$$

Control law

$$\boxed{\text{if } \mathcal{L}_g h \neq 0} \qquad u = \frac{1}{\mathcal{L}_g h}\left(-\mathcal{L}_f h + \underbrace{\dot{y}^{\text{des}} + k(y^{\text{des}} - y)}_{v}\right)$$

Closed loop system

$$\dot{y} - \underbrace{\dot{y}^{\text{des}} + k(y - y^{\text{des}})}_{v} = 0 \qquad \Rightarrow \qquad \dot{y} = v$$

Input-Output Linearization

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

Output $y = h(x)$

Rate of change of output

$$\dot{y} = \mathcal{L}_f h + \cancel{(\mathcal{L}_g h)}\, u$$

Control law

$$\text{if } \mathcal{L}_g h \neq 0 \qquad u = \frac{1}{\mathcal{L}_g h}\left(-\mathcal{L}_f h + \dot{y}^{\text{des}} + k(y^{\text{des}} - y)\right)$$

$$\boxed{\text{if } \mathcal{L}_g h = 0} \qquad \dot{y} = \mathcal{L}_f h$$

(rate of change of output is independent of u)

Explore higher order derivatives of output

nonzero ?

$$\ddot{y} = \mathcal{L}_f \mathcal{L}_f h + \boxed{(\mathcal{L}_g \mathcal{L}_f h)}\, u$$

Affine SISO System

State $\mathbf{x} \in R^n$

Input $u \in R$

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

Output $y = h(\mathbf{x}) \in R$

Relative degree r

- The index of the first nonzero term in the sequence

$$\mathcal{L}_f^2 h = \mathcal{L}_f\left(\mathcal{L}_f h\right)$$

$$\mathcal{L}_f^3 h = \mathcal{L}_f\left(\mathcal{L}_f\left(\mathcal{L}_f h\right)\right)$$

…

$$\underbrace{\mathcal{L}_g h}_{r=1}, \underbrace{\mathcal{L}_g \mathcal{L}_f h}_{r=2}, \mathcal{L}_g \mathcal{L}_f^2 h, \ldots, \overbrace{\mathcal{L}_g \mathcal{L}_f^k h}^{r=k+1}, \ldots,$$

Affine SISO System

□ *Linear control, model independent*

↑ *feed forward*

↑ *feedback*

$r = 1$

$$u = \boxed{\frac{1}{\mathcal{L}_g h}}\left(-\mathcal{L}_f h + \boxed{\underset{\uparrow}{\dot{y}^{\text{des}}} + k(\underset{\uparrow}{y^{\text{des}}} - \underset{\uparrow}{y})}\right)$$

$r = 2$

$$u = \boxed{\frac{1}{\mathcal{L}_g \mathcal{L}_f h}}\left(-\mathcal{L}_f \mathcal{L}_f h + \boxed{\underset{\uparrow}{\ddot{y}^{\text{des}}} + k_1(\underset{\uparrow}{\dot{y}^{\text{des}}} - \underset{\uparrow}{\dot{y}}) + k_2(\underset{\uparrow}{y^{\text{des}}} - \underset{\uparrow}{y})}\right)$$

$r = 3$

$$u = \boxed{\frac{1}{\mathcal{L}_g \mathcal{L}_f^2 h}}\left(-\mathcal{L}_f^3 h + \boxed{\underset{\uparrow}{\dddot{y}^{\text{des}}} + k_1(\underset{\uparrow}{\ddot{y}^{\text{des}}} - \underset{\uparrow}{\ddot{y}}) + k_2(\underset{\uparrow}{\dot{y}^{\text{des}}} - \underset{\uparrow}{\dot{y}}) + k_3(\underset{\uparrow}{y^{\text{des}}} - \underset{\uparrow}{y})}\right)$$

General form of control law

$$u = \alpha(x) + \boxed{\beta(x)}\boxed{v}$$

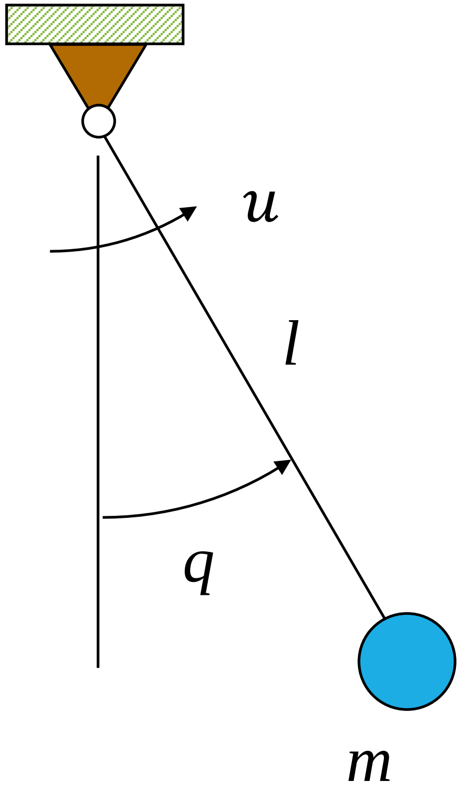
Example

$$ml^2\ddot{q} + mgl\sin q = u$$

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} q \\ \dot{q} \end{bmatrix}$$

$$\dot{x} = \underbrace{\boxed{\begin{bmatrix} x_2 \\ -\frac{g}{l}\sin(x_1) \end{bmatrix}}}_{f(x)} + \underbrace{\boxed{\begin{bmatrix} 0 \\ \frac{1}{ml^2} \end{bmatrix}}}_{g(x)} u$$

$$h = x_1$$



$$\mathcal{L}_g h = 0 \qquad \mathcal{L}_f h = x_2$$

$$\boxed{\mathcal{L}_g\mathcal{L}_f h = \frac{1}{ml^2}} \qquad \mathcal{L}_f^2 h = -\frac{g}{l}\sin x_1$$

r=2

$$\boxed{u = \frac{1}{\mathcal{L}_g\mathcal{L}_f h}\left(-\mathcal{L}_f\mathcal{L}_f h + \ddot{y}^{\mathrm{des}} + k_1(\dot{y}^{\mathrm{des}} - \dot{y}) + k_2(y^{\mathrm{des}} - y)\right)}$$

Affine MIMO System

State $\mathbf{x} \in R^n$

Input $u \in R^m$

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

Output $y = h(\mathbf{x}) \in R^n$

Assume each output has relative degree r

Nonlinear feedback law

$$u = \left(\mathcal{L}_g \mathcal{L}_f^{r-1} h\right)^{-1} \left(-\mathcal{L}_f^r + v\right)$$

Leads to equivalent system

$$y^{(r)} = v$$

Example: Robot Arm

Fully-actuated robot (n joints, n actuators)

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + N(q) = \tau$$

Dynamic model

- M is the positive definite, $n \times n$ inertia matrix
- C is the $n \times n$ matrix of Coriolis and centripetal forces
- N is the n-dimensional vector of gravitational forces
- τ is the n-dimensional vector of actuator forces and torques

Key: M is non singular

Example: Robot Arm

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + N(q) = \tau$$

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} q \\ \dot{q} \end{bmatrix} \qquad u = \tau \in \mathbb{R}^n$$

$$\dot{x} = \begin{bmatrix} x_2 \\ -M(x_1)^{-1}(N(x_1) + C(x_1, x_2)x_2) \end{bmatrix} + \begin{bmatrix} 0 \\ M(x_1)^{-1} \end{bmatrix} u$$

$$h(x) = x_1$$

Example: Robot Arm

$$f(x) = \begin{bmatrix} x_2 \\ -M(x_1)^{-1}(N(x_1) + C(x_1, x_2)x_2) \end{bmatrix} \quad g(x) = \begin{bmatrix} 0 \\ M(x_1)^{-1} \end{bmatrix}$$

$$h(x) = x_1$$

$$\mathcal{L}_g h = 0,\ \mathcal{L}_g \mathcal{L}_f h \neq 0$$

Relative degree is 2

$$u = \boxed{(\mathcal{L}_g \mathcal{L}_f h)^{-1}} \left(\boxed{-\mathcal{L}_f \mathcal{L}_f h} + \boxed{\ddot{y}^{\text{des}} + k_1(\dot{y}^{\text{des}} - \dot{y}) + k_2(y^{\text{des}} - y)}\right)$$

Control law

$$u = \boxed{M(x_1)}\left(\boxed{M(x_1)^{-1}(N(x_1) + C(x_1, x_2)x_2)} + \boxed{\ddot{y}^{des} + k_1(\dot{y}^{des} - \dot{y}) + k_2(y^{des} - y)}\right)$$

Kinematic Cart

State equations, inputs

$$\dot{x} = v\cos\theta$$
$$\dot{y} = v\sin\theta$$
$$\dot{\theta} = \omega$$

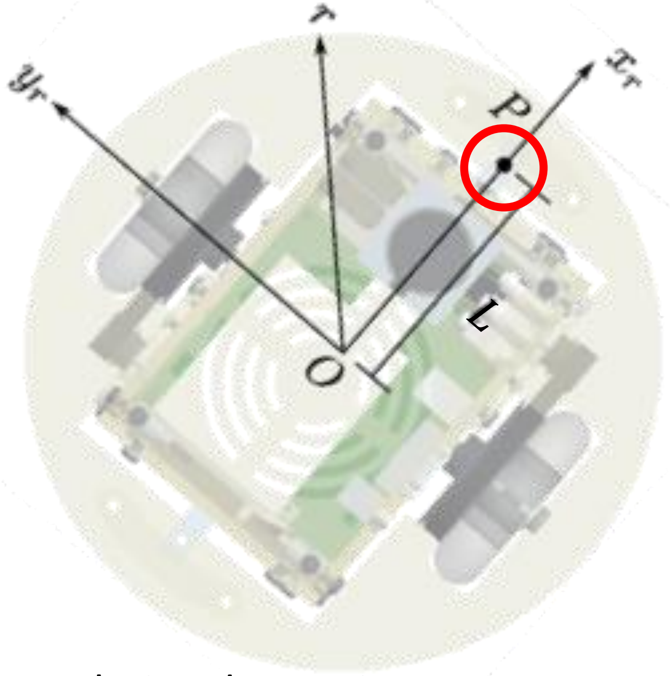
$$\dot{X} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}$$

$$\dot{X} = g(X)u$$

Outputs

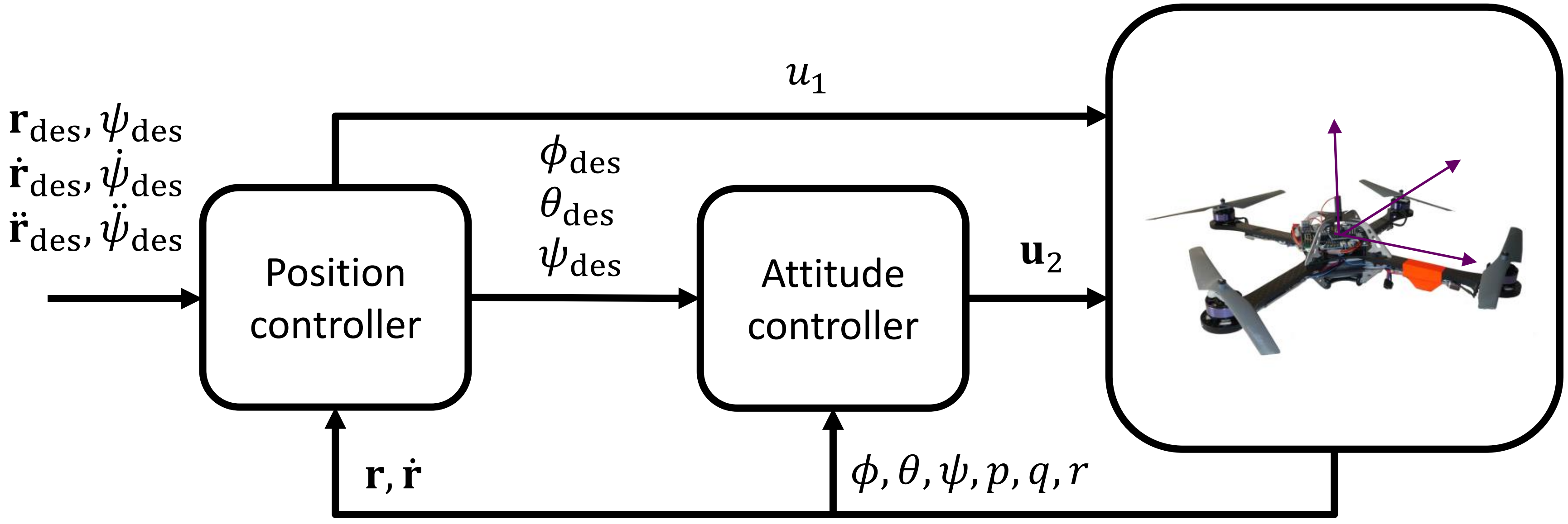
$$y = h(x) = x_P = \begin{bmatrix} x + L\cos\theta \\ x + L\sin\theta \end{bmatrix}$$

$$\dot{y} = \begin{bmatrix} \cos\theta & -L\sin\theta \\ \sin\theta & L\cos\theta \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}$$



Relative degree is 1

Recall Linear Quadrotor Control


$\mathbf{r}_{\text{des}}, \psi_{\text{des}}$
$\dot{\mathbf{r}}_{\text{des}}, \dot{\psi}_{\text{des}}$
$\ddot{\mathbf{r}}_{\text{des}}, \ddot{\psi}_{\text{des}}$
Position controller
u_1
ϕ_{des}
θ_{des}
ψ_{des}
Attitude controller
$\mathbf{u}_2$
$\mathbf{r}, \dot{\mathbf{r}}$
$\phi, \theta, \psi, p, q, r$

1) PD controller on position generated commanded accelerations $\ddot{\boldsymbol{r}}_c$

2) Using $\ddot{\boldsymbol{r}}_c$, find u_1 and desired roll/pitch

$$u_1 = m(g + \ddot{r}_{3,c})$$

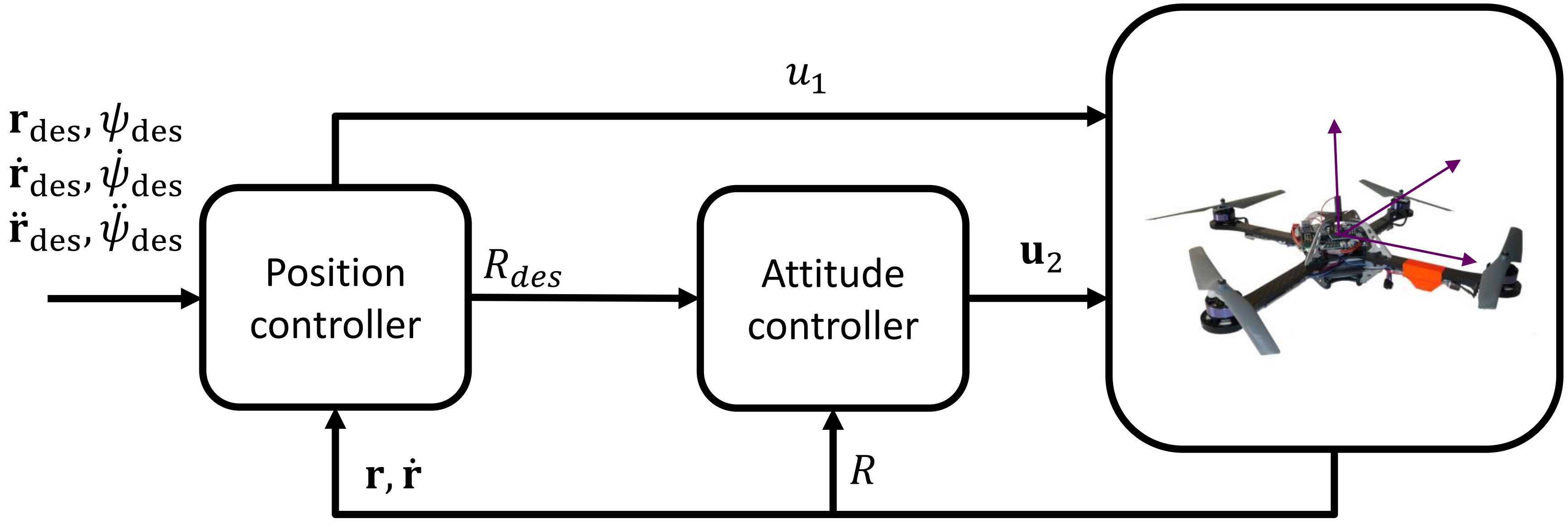
$$\phi_c = \frac{1}{g}(\ddot{r}_{1,c} \sin \psi_{des} - \ddot{r}_{2,c} \cos \psi_{des})$$

$$\theta_c = \frac{1}{g}(\ddot{r}_{1,c} \cos \psi_{des} + \ddot{r}_{2,c} \sin \psi_{des})$$

3) Using desired roll/pitch, calculate $\boldsymbol{u}_2$

$$\mathbf{u}_2 = I \begin{bmatrix} k_{d,\phi}(p_{\text{des}} - p) + k_{p,\phi}(\phi_{\text{des}} - \phi) \\ k_{d,\theta}(q_{\text{des}} - q) + k_{p,\theta}(\theta_{\text{des}} - \theta) \\ k_{d,\psi}(r_{\text{des}} - r) + k_{p,\psi}(\psi_{\text{des}} - \psi) \end{bmatrix}$$

Geometric Tracking Control


u_1
$\mathbf{r}_{\text{des}}, \psi_{\text{des}}$
$\dot{\mathbf{r}}_{\text{des}}, \dot{\psi}_{\text{des}}$
$\ddot{\mathbf{r}}_{\text{des}}, \ddot{\psi}_{\text{des}}$
Position controller
R_{des}
Attitude controller
$\mathbf{u}_2$
$\mathbf{r}, \dot{\mathbf{r}}$
R

- How do I choose R_{des}?
  - Point thrust in the direction of $\ddot{\boldsymbol{r}}_c$
  - Specify yaw angle from ψ_{des}

- How can I do attitude control on rotation matrices?
  - What is a “rotation error”?

Planar Quadrotor

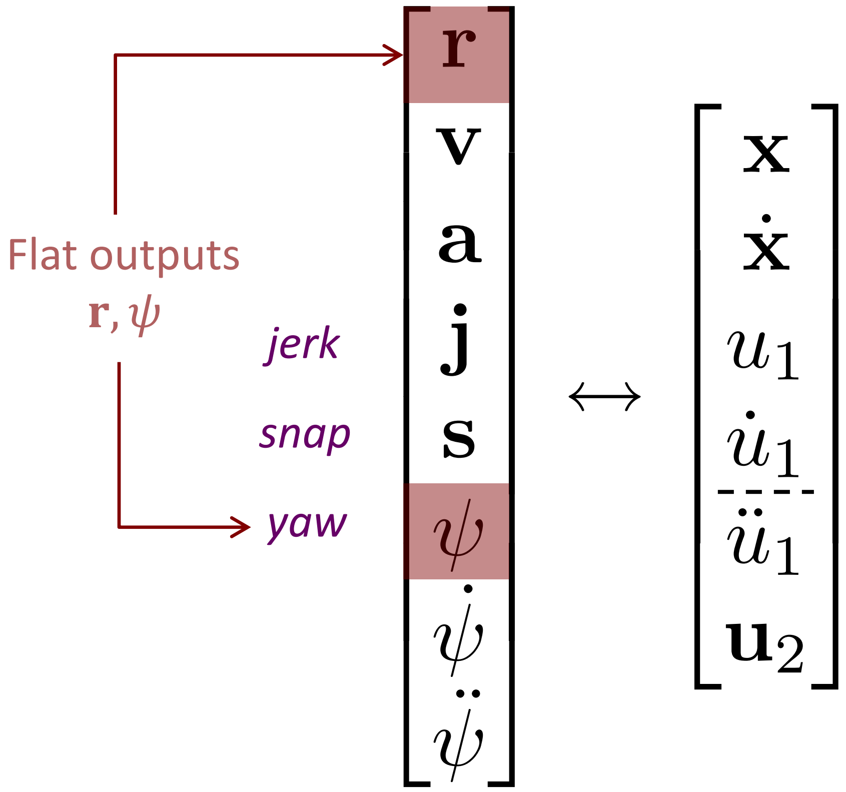
The planar quadrotor is a differentially flat system

- All state variables and the inputs can be written as smooth functions of *flat outputs* and their derivatives

$$\begin{bmatrix} \boxed{\begin{matrix} y \\ z \end{matrix}} \\ \dot{y} \\ \dot{z} \\ \ddot{y} \\ \ddot{z} \\ \dddot{y} \\ \dddot{z} \\ y^{(iv)} \\ z^{(iv)} \end{bmatrix} \quad \overset{\text{diffeomorphism}}{\longleftrightarrow} \quad \begin{array}{l} \left.\begin{matrix} y \\ z \\ \phi \\ \dot{y} \\ \dot{z} \\ \dot{\phi} \\ u_1 \\ \dot{u}_1 \end{matrix}\right\} \text{State} \\ \hdashline \left.\begin{matrix} \ddot{u}_1 \\ u_2 \end{matrix}\right\} \text{Inputs} \end{array}$$

3D Quadrotor

State $[\mathbf{x}, \dot{\mathbf{x}}]$ where $\mathbf{x} = [x, y, z, \phi, \theta, \psi]^T$



$$\begin{bmatrix} \mathbf{r} \\ \mathbf{v} \\ \mathbf{a} \\ \mathbf{j} \\ \mathbf{s} \\ \psi \\ \dot{\psi} \\ \ddot{\psi} \end{bmatrix} \leftrightarrow \begin{bmatrix} \mathbf{x} \\ \dot{\mathbf{x}} \\ u_1 \\ \dot{u}_1 \\ \ddot{u}_1 \\ \mathbf{u}_2 \end{bmatrix}$$

[Mellinger and Kumar, ICRA 2011]

Other References

Canvas Course Notes: Input-Output Linearization

Quadrotor Control:

Daniel Mellinger and Vijay Kumar, "Minimum snap trajectory generation and control for quadrotors", 2011.

Taeyoung Lee, Melvin Leoky, and N. Harris McClamroch. "Geometric tracking control of a quadrotor UAV on SE (3)," 2010.

Other Resources:

Nonlinear Systems: Analysis, Stability, and Control by Shankar Sastry

Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation Ch 1, by Russ Tedrake.