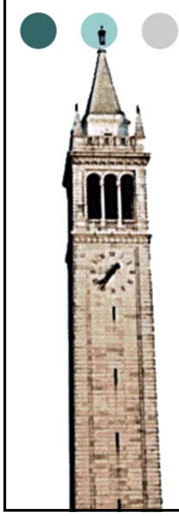


Introduction to Embedded Systems

Sanjit A. Seshia

UC Berkeley

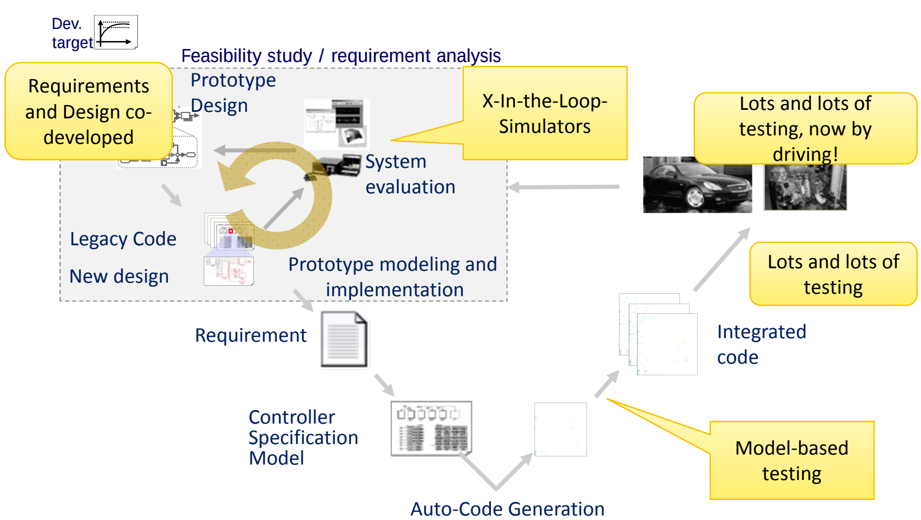
EECS 149/249A

Fall 2015

Chapter 2: Model Based Design

Model-Based Design in Industry: An example – the so-called "V" model

[from J. Deshmukh, Toyota]



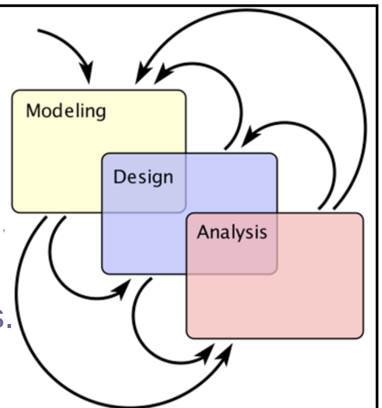
Our Approach in this Course

Mathematical Models are Central to the Design Process

Create models, and use them judiciously!

But there is no unique approach to Model-Based Design (at least today).

Modeling, Design, Analysis: An Iterative Process



Modeling is the process of gaining a deeper understanding of a system through imitation. Models specify **what** a system does.

Design is the structured creation of artifacts. It specifies **how** a system does what it does. This includes optimization.

Analysis is the process of gaining a deeper understanding of a system through dissection. It specifies **why** a system does what it does (or fails to do what a model says it should do).

Modeling Techniques in this Course

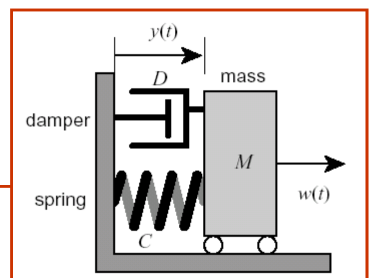
Models that are abstractions of **system dynamics** (how system behavior changes over time)

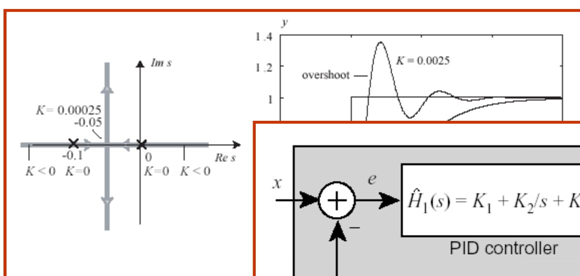
Examples:

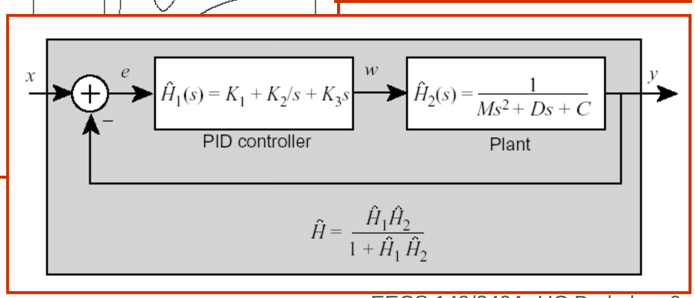
- Modeling physical phenomena – differential eqns
- Feedback control systems – time-domain modeling
- Modeling modal behavior – FSMs, hybrid automata, …
- Modeling sensors and actuators – models that help with calibration, noise elimination, …
- Modeling hardware and software – capture concurrency, timing, power, …
- Modeling networks – latencies, error rates, packet loss, …

Today's Lecture: Modeling of Continuous Dynamics

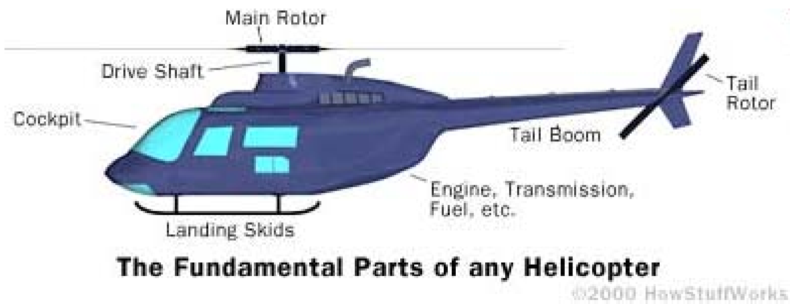
Ordinary differential equations, Laplace transforms, feedback control models, …







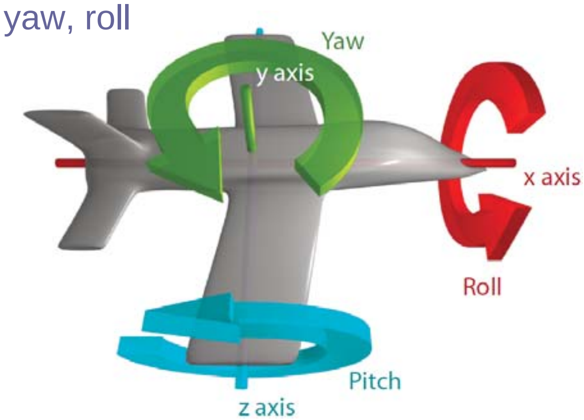
An Example: Modeling Helicopter Dynamics



Modeling Physical Motion

Six degrees of freedom:

- Position: x, y, z
- Orientation: pitch, yaw, roll



$f: A \to B$ $\quad f \subset A \times B \Rightarrow f(t)$ is not a function, while $f(t)$ is

$A = \mathbb{R},\ t \in \mathbb{R},\ f(t) \in B$

Notation

Position is given by three functions:

$$x : \mathbb{R} \to \mathbb{R}$$
$$y : \mathbb{R} \to \mathbb{R}$$
$$z : \mathbb{R} \to \mathbb{R}$$

where the domain $\mathbb{R}$ represents time and the co-domain (range) $\mathbb{R}$ represents position along the axis. Collecting into a vector:

$$\mathbf{x} : \mathbb{R} \to \mathbb{R}^3$$

Position at time $t \in \mathbb{R}$ is $\mathbf{x}(t) \in \mathbb{R}^3$.

Notation

Velocity

$$\dot{\mathbf{x}} : \mathbb{R} \to \mathbb{R}^3$$

is the derivative, $\forall\, t \in \mathbb{R}$,

$$\boxed{\dot{\mathbf{x}}(t) = \frac{d}{dt}\mathbf{x}(t)} \quad x'(t) = \frac{d}{dt}x(t)$$

Acceleration $\ddot{\mathbf{x}} : \mathbb{R} \to \mathbb{R}^3$ is the second derivative,

$$\boxed{\ddot{\mathbf{x}} = \frac{d^2}{dt^2}\mathbf{x}}$$

$F = ma$

Force on an object is $\mathbf{F} : \mathbb{R} \to \mathbb{R}^3$.

$L = 7B$

$S(t) = \|x'(t)\| = \sqrt{V_1^2(t) + V_2^2(t) + V_3^2(t)}$ while $x'(t) = (V_1, V_2, V_3)$

Newton's Second Law

Newton's second law states $\forall\, t \in \mathbb{R}$,

$$\boxed{\mathbf{F}(t) = M\ddot{\mathbf{x}}(t)}$$

$F(t) = M\ddot{x}(t)$

where M is the mass. To account for initial position and velocity, convert this to an integral equation

$$\begin{aligned} \mathbf{x}(t) &= \mathbf{x}(0) + \int_0^t \dot{\mathbf{x}}(\tau)d\tau \\ &= \mathbf{x}(0) + t\dot{\mathbf{x}}(0) + \frac{1}{M}\int_0^t \int_0^\tau \mathbf{F}(\alpha)d\alpha d\tau, \end{aligned}$$

Maybe function of time/position

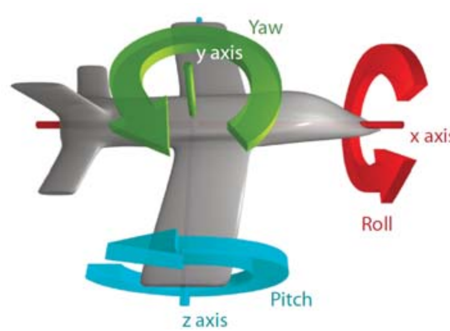
⇓

$F(t, x, y, z)$

or $F(t, \underline{x})$ ← Not good mode!

$\underline{x}: \mathbb{R} \to \mathbb{R}^3$

Orientation



- Orientation: $\theta\colon \mathbb{R} \to \mathbb{R}^3$
- Angular velocity: $\dot{\theta}\colon \mathbb{R} \to \mathbb{R}^3$
- Angular acceleration: $\ddot{\theta}\colon \mathbb{R} \to \mathbb{R}^3$
- Torque: $\mathbf{T}\colon \mathbb{R} \to \mathbb{R}^3$

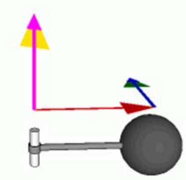
扭矩

$$\theta(t) = \begin{bmatrix} \theta_x(t) \\ \theta_y(t) \\ \theta_z(t) \end{bmatrix} = \begin{bmatrix} \text{roll} \\ \text{yaw} \\ \text{pitch} \end{bmatrix}$$

Angular version of force is torque.
For a point mass rotating around a fixed axis:

- radius of the arm: $r \in \mathbb{R}$
- force orthogonal to arm: $f \in \mathbb{R}$
- mass of the object: $m \in \mathbb{R}$

$M = J\beta$



$$T_y(t) = r f(t)$$

angular momentum, momentum

Just as force is a push or a pull, a torque is a twist.
Units: newton-meters/radian, Joules/radian

Note that radians are meters/meter (2π meters of circumference per 1 meter of radius), so as units, are optional.

$\dot{\theta} \Rightarrow$ 角速度

Rotational Version of Newton's Second Law

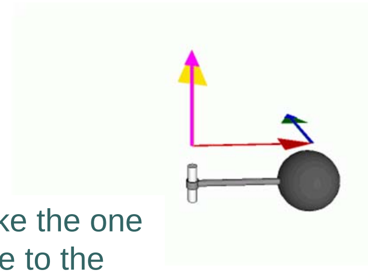
$$\mathbf{T}(t) = \frac{d}{dt}\left(I(t)\dot{\theta}(t)\right),$$

where $I(t)$ is a 3×3 matrix called the moment of inertia tensor.

$$\begin{bmatrix} T_x(t) \\ T_y(t) \\ T_z(t) \end{bmatrix} = \frac{d}{dt}\left(\begin{bmatrix} I_{xx}(t) & I_{xy}(t) & I_{xz}(t) \\ I_{yx}(t) & I_{yy}(t) & I_{yz}(t) \\ I_{zx}(t) & I_{zy}(t) & I_{zz}(t) \end{bmatrix}\begin{bmatrix} \dot{\theta}_x(t) \\ \dot{\theta}_y(t) \\ \dot{\theta}_z(t) \end{bmatrix}\right)$$

Here, for example, $T_y(t)$ is the net torque around the y axis (which would cause changes in yaw), $I_{yx}(t)$ is the inertia that determines how acceleration around the x axis is related to torque around the y axis.

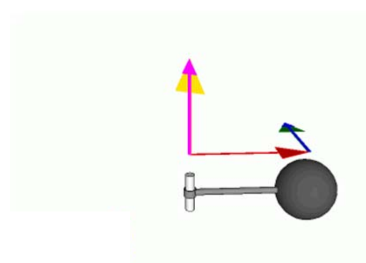
Feedback Control Problem



A helicopter without a tail rotor, like the one below, will spin uncontrollably due to the torque induced by friction in the rotor shaft.

Control system problem:
Apply torque using the tail rotor to counterbalance the torque of the top rotor.

Simplified Model



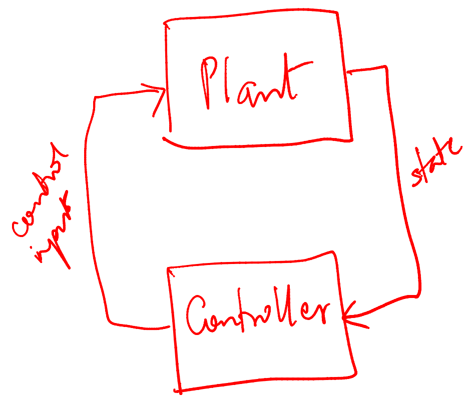
Yaw dynamics:

$$T_y(t) = I_{yy}\ddot{\theta}_y(t)$$

To account for initial angular velocity, write as

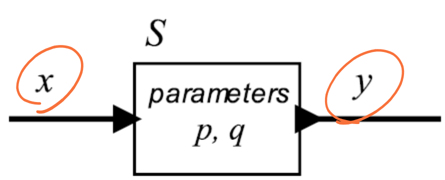
$$\boxed{\dot{\theta}_y(t) = \dot{\theta}_y(0) + \frac{1}{I_{yy}} \int_0^t T_y(\tau)d\tau.}$$

“Plant” and Controller



Actor Model of Systems

A *system* is a function that accepts an input *signal* and yields an output signal.



The domain and range of the system function are sets of signals, which themselves are functions.

$$x\colon \mathbb{R} \to \mathbb{R}, \quad y\colon \mathbb{R} \to \mathbb{R}$$

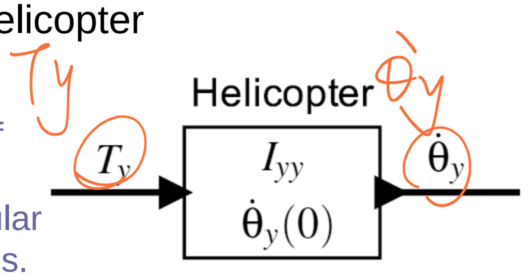
$$S\colon X \to Y$$

$$X = Y = (\mathbb{R} \to \mathbb{R})$$

Parameters may affect the definition of the function *S*.

Actor Model of the Helicopter

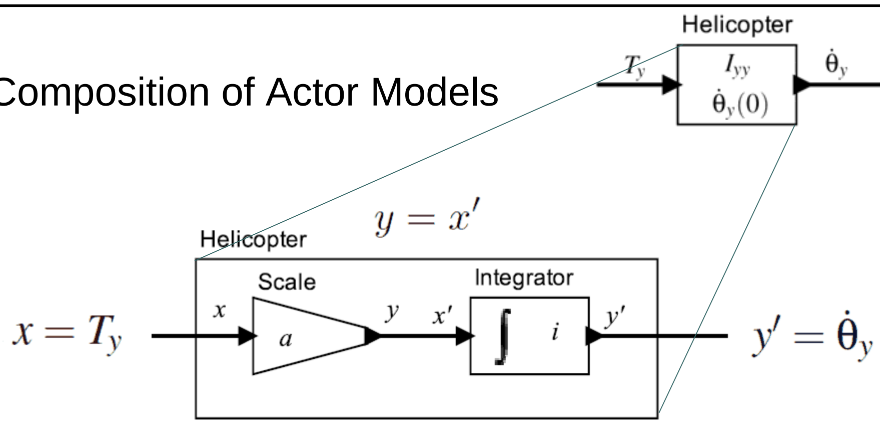
Input is the net torque of the tail rotor and the top rotor. Output is the angular velocity around the *y* axis.



Parameters of the model are shown in the box. The input and output relation is given by the equation to the right.

$$\dot{\theta}_y(t) = \dot{\theta}_y(0) + \frac{1}{I_{yy}} \int_0^t T_y(\tau)d\tau$$

Composition of Actor Models



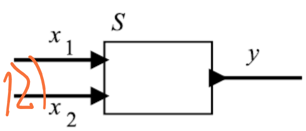
$$\forall\, t \in \mathbb{R}, \quad y(t) = ax(t) \qquad y'(t) = i + \int_0^t x'(\tau)d\tau$$

$$y = ax$$

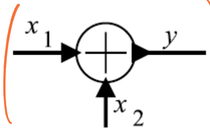
$$a = 1/I_{yy} \qquad i = \dot{\theta}_y(0)$$

$S : (\mathbb{R} \to \mathbb{R}) \to (\mathbb{R} \to \mathbb{R})$

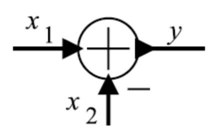
Actor Models with Multiple Inputs



$$S : (\mathbb{R} \to \mathbb{R})^2 \to (\mathbb{R} \to \mathbb{R})$$



$$\forall\, t \in \mathbb{R}, \quad y(t) = x_1(t) + x_2(t)$$



$$(S(x_1, x_2))(t) = y(t) = x_1(t) - x_2(t)$$

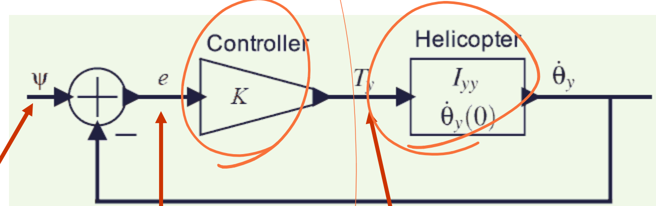
$e(t) = \psi(t) - \dot{\theta}_y(t)$

$\Rightarrow T_y = Ke(t)$

$\Downarrow$

$\dot{\theta}_y(t) =$

Proportional controller



desired angular velocity

error signal

net torque

$$e(t) = \psi(t) - \dot{\theta}_y(t)$$

$$T_y(t) = Ke(t)$$

$$\dot{\theta}_y(t) = \dot{\theta}_y(0) + \frac{1}{I_{yy}} \int_0^t T_y(\tau)\, d\tau$$

$$= \dot{\theta}_y(0) + \frac{K}{I_{yy}} \int_0^t (\psi(\tau) - \dot{\theta}_y(\tau))\, d\tau$$

Note that the angular velocity appears on both sides, so this equation is not trivial to solve.

$\Rightarrow \dot{\theta}_y(t) = \dot{\theta}_y(0) e^{-\frac{K}{I_{yy}} t} u(t)$

$\dot{\theta}_y(t)\big|_{t=0} = \dot{\theta}_y(0)$ $C_2 = \dot{\theta}_y(0)$

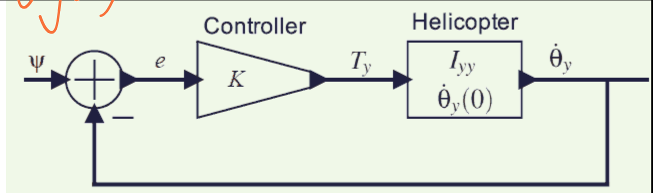
$\Rightarrow \dot{\theta}_y(t) = C_2 e^{-\frac{K}{I_{yy}} t}$

$\theta_y(t) = C_1 + C_2 e^{-\frac{K}{I_{yy}} t}$

$\Rightarrow \ddot{\theta}_y(t) = -\frac{K}{I_{yy}} \dot{\theta}_y(t)$

$\ddot{\theta}_y(t) = 0 - \frac{K}{I_{yy}} \dot{\theta}_y(t)$

Behavior of the controller



$$\dot{\theta}_y(t) = \dot{\theta}_y(0) + \frac{K}{I_{yy}} \int_0^t (\psi(\tau) - \dot{\theta}_y(\tau)) d\tau$$

Desired angular velocity: $\psi(t) = 0$

Simplifies differential equation to:

$$\dot{\theta}_y(t) = \dot{\theta}_y(0) - \frac{K}{I_{yy}} \int_0^t \dot{\theta}_y(\tau) d\tau$$

Which can be solved as follows (see textbook):

$$\dot{\theta}_y(t) = \dot{\theta}_y(0) e^{-Kt/I_{yy}} u(t)$$

Questions

- Can the behavior of this controller change when it is implemented in software?
- How do we measure the angular velocity in practice? How do we incorporate noise into this model?