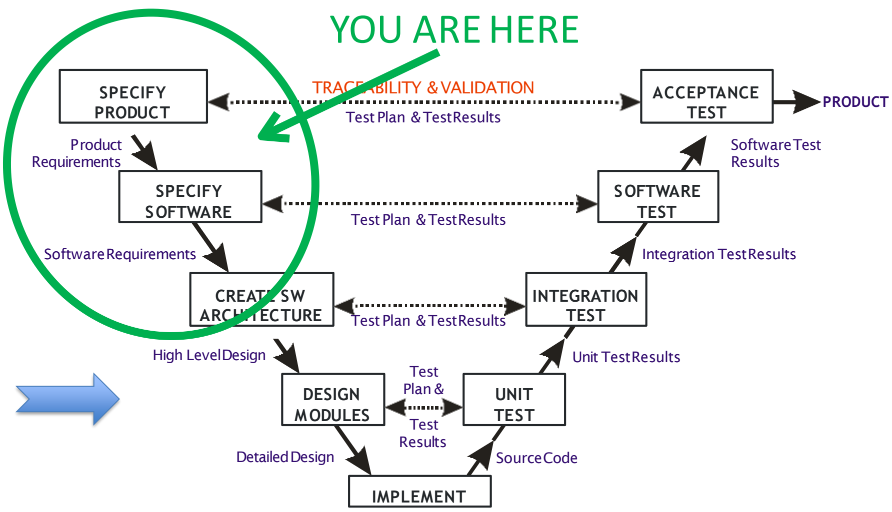
Lecture 4: Modeling Physical Dynamics

Seyed-Hosein Attarzadeh-Niaki

Based on the Slides by Edward Lee

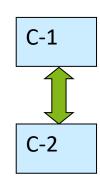
Review

- CPS requirements
  - Functional requirements
  - Extra-functional requirements
    - Real-time-ness
    - Efficiency (energy, code size, run time, etc.)
    - Dependability
- Requirement analysis



Models of Computation

- **What does it mean, "to compute"?**
- **Models of computation define**
  - Components and an execution model for computations of each component
  - Communication model for exchange of information between components.



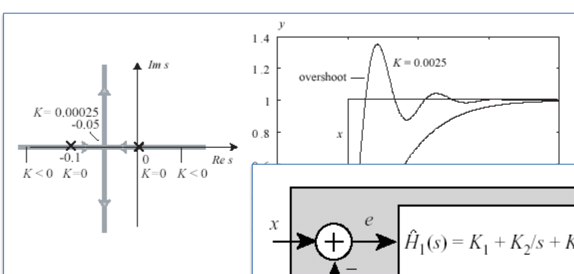
Modeling Techniques in this Course

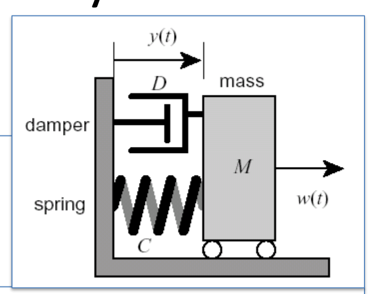
Models that are abstractions of **system dynamics** (how system behavior changes over time)

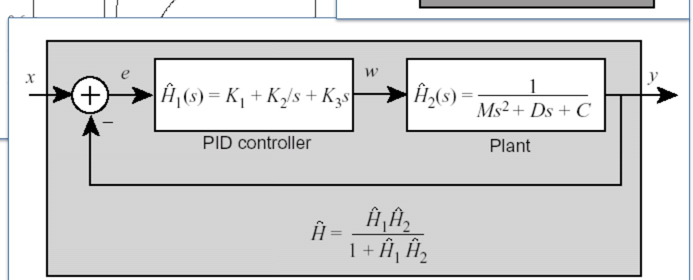
- Modeling continuous dynamics – differential equations
  - Feedback control systems – time-domain modeling
- Modeling discrete dynamics – finite-state machines
- Modeling hybrid systems – modal models, timed automata
- Concurrent models of computation
  - Synchronous composition
  - Dataflow models
  - …

Modeling Continuous Dynamics

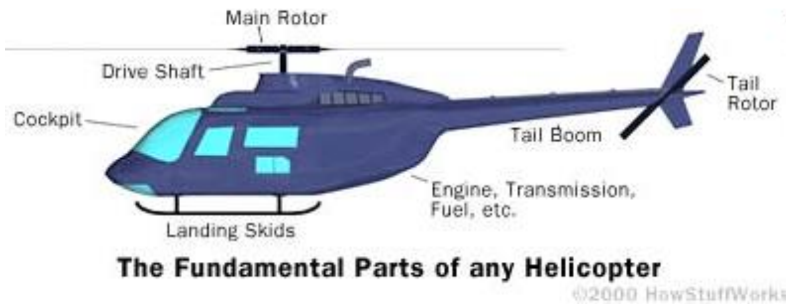
Ordinary differential equations, Laplace transforms, feedback control models, …







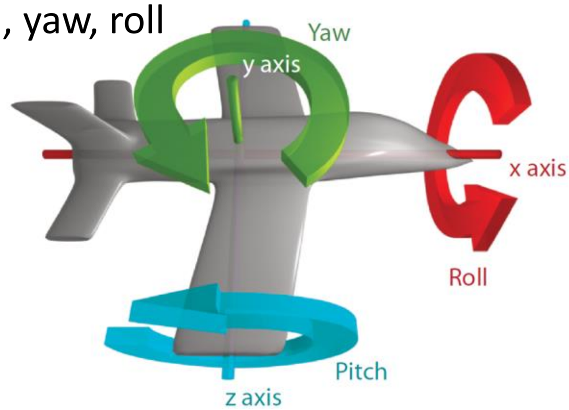
An Example: Helicopter Dynamics



Modeling Physical Motion

Six degrees of freedom

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



Notation
Continuous-Time Signals

Position is given by three functions:

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

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

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

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

Notation
Differential Equation

Velocity

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

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

$$\dot{\mathbf{x}}(t) = \frac{d}{dt}\mathbf{x}(t)$$

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

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

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

Newton's Second Law Integral Equations

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

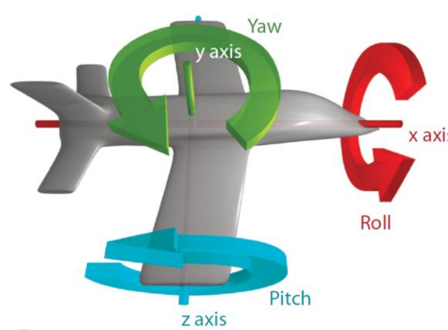
$$\mathbf{F}(t) = M\ddot{\mathbf{x}}(t)$$

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

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

Orientation

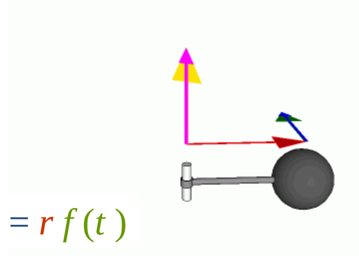
- Orientation: $\theta\colon \mathbb{R} \to \mathbb{R}^3$
- Angular velocity: $\dot{\theta}\colon \mathbb{R} \to \mathbb{R}^3$
- Angular acceleration: $\ddot{\theta}\colon \mathbb{R} \to \mathbb{R}^3$
- Torque: $\mathbf{T}\colon \mathbb{R} \to \mathbb{R}^3$



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

Angular Version of Force: 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: $m \in \mathbb{R}$



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

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.

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.

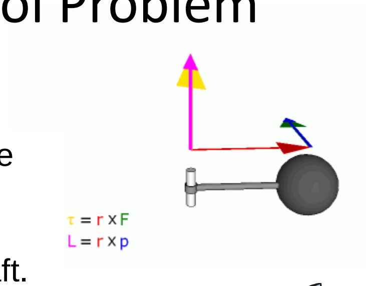
$$\begin{bmatrix} T_x(t) \\ T_y(t) \\ T_z(t) \end{bmatrix} = \frac{d}{dt}\left(\begin{bmatrix} I_{xx}(t) & I_{xy}(t) & I_{xz}(t) \\ I_{yx}(t) & I_{yy}(t) & I_{yz}(t) \\ I_{zx}(t) & I_{zy}(t) & I_{zz}(t) \end{bmatrix}\begin{bmatrix} \dot{\theta}_x(t) \\ \dot{\theta}_y(t) \\ \dot{\theta}_z(t) \end{bmatrix}\right)$$

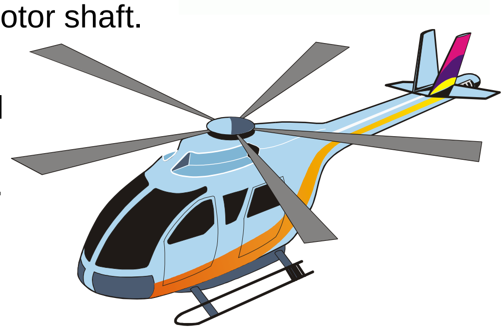
Here, for example, $T_y(t)$ is the net torque around the y axis (which would cause changes in yaw), $I_{yx}(t)$ is the inertia that determines how acceleration around the x axis is related to torque around the y axis.

Feedback Control Problem

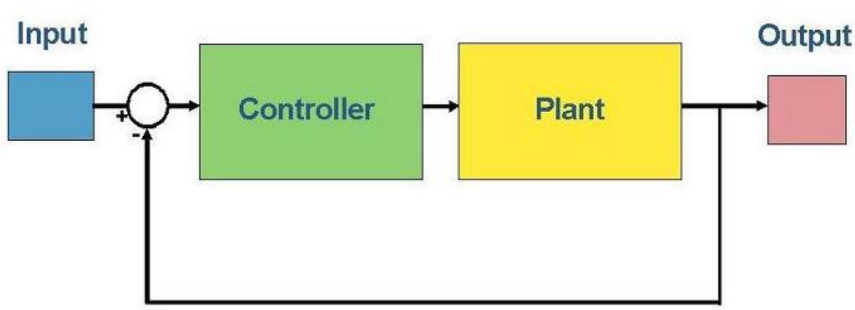
A helicopter without a tail rotor, like the one below, will *spin uncontrollably* due to the torque induced by friction in the rotor shaft.

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

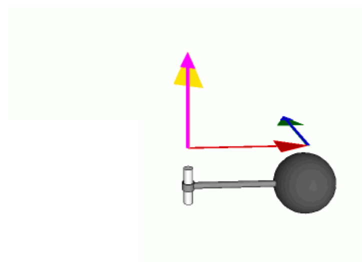




Plant and Controller



Simplified Model



Yaw dynamics:

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

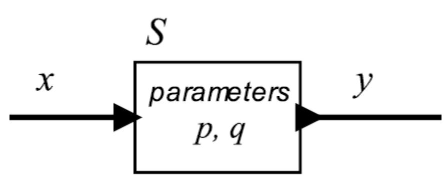
To account for initial angular velocity, write as

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

This type of simplification is called "model order reduction".

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.

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

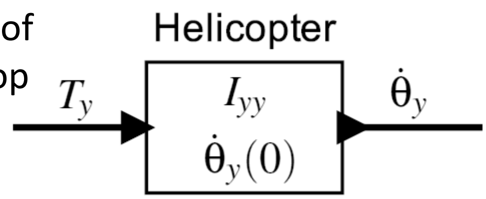
$$S: X \to Y$$

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

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

Actor Model of the Helicopter

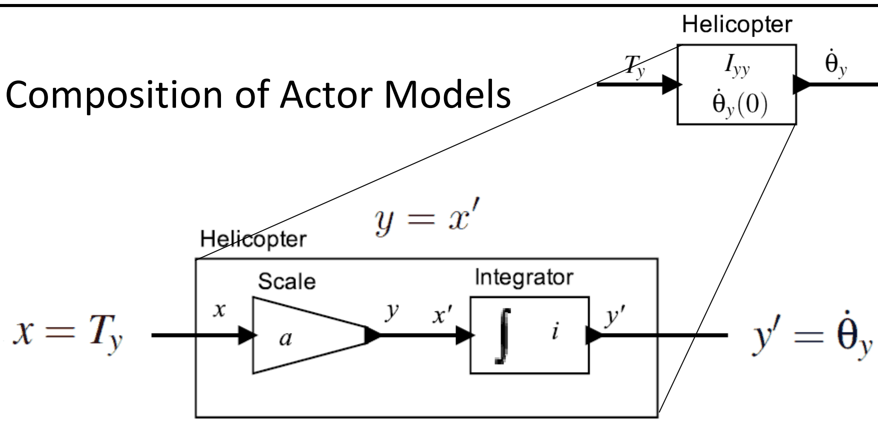
- Input is the net torque of the tail rotor and the top rotor.
- Output is the angular velocity around the *y* axis.



- 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

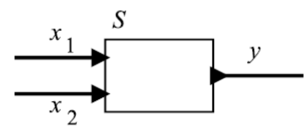


$$\forall\, t \in \mathbb{R}, \quad y(t) = ax(t) \qquad y'(t) = i + \int_0^t x'(\tau)d\tau$$

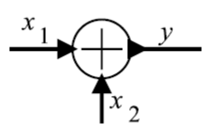
$$y = ax$$

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

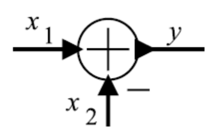
Actor Models with Multiple Inputs



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

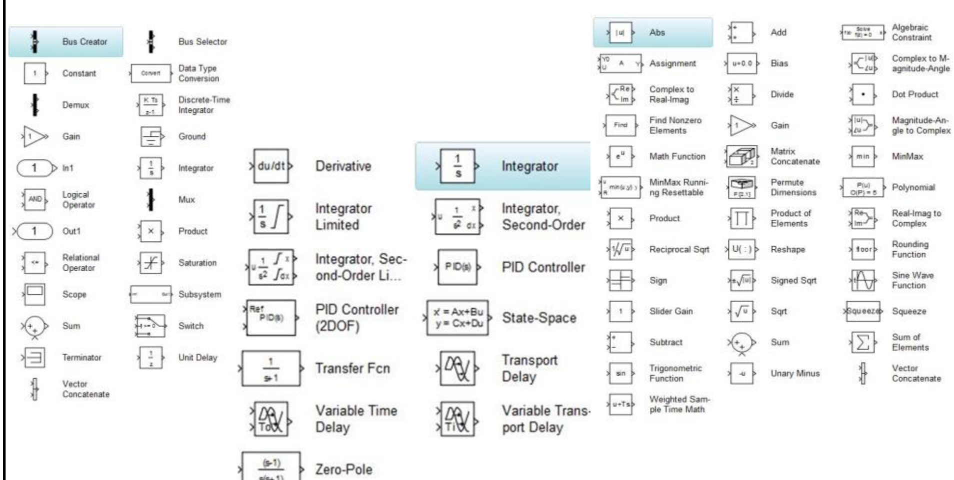


$$\forall\, t \in \mathbb{R}, \quad y(t) = x_1(t) + x_2(t)$$

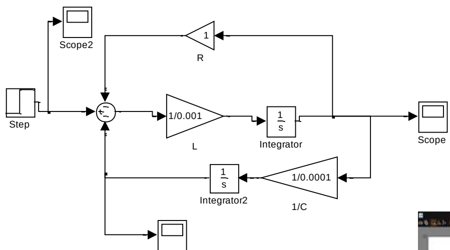


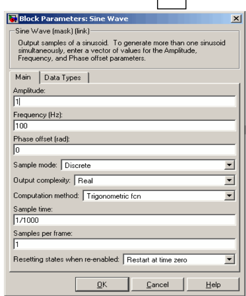
$$(S(x_1, x_2))(t) = y(t) = x_1(t) - x_2(t)$$

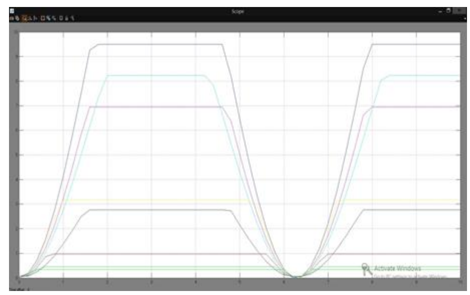
Simulink Library



Simulink Models







Properties of Systems I

Causal systems

- A system is causal if its output depends only on current and past inputs
  - i.e., if for two possible inputs that are identical up to (and including) time τ, the outputs are identical up to (and including) time τ.

Memoryless systems

- A system has memory if the output depends not only on the current inputs, but also on past inputs

Properties of Systems II

Linear and time-invariant (LTI) systems

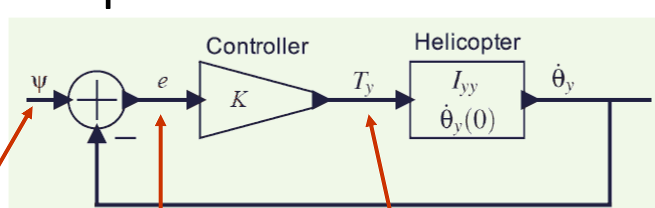
- Satisfy superposition
  - ∀ $x_1,x_2 \in X$ and ∀ $a,b \in R$, $\mathbf{S(ax_1+bx_2) = aS(x_1) + bS(x_2)}$
- Time invariance ($\mathbf{D_\tau}$ is the delay actor)
  - ∀ $x \in X$ and ∀ $\tau \in R$, $\mathbf{S(D_\tau(x)) = D_\tau(S(x))}$ if $(D_\tau(x))(t) = x(t-\tau)$

Stable systems

- A system is said to be stable if the output signal is bounded for all input signals that are bounded
  - Bounded-input bounded-output stable (BIBO stable)

Check LeeSeshia for formal definitions.

Proportional Controller



desired angular velocity

error signal

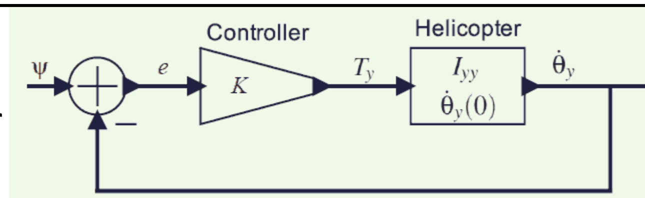
net torque

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

$$\begin{aligned}\dot{\theta}_y(t) &= \dot{\theta}_y(0) + \frac{1}{I_{yy}}\int_0^t T_y(\tau)d\tau \\ &= \dot{\theta}_y(0) + \frac{K}{I_{yy}}\int_0^t (\psi(\tau) - \dot{\theta}_y(\tau))d\tau\end{aligned}$$

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

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):

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

Exercise

- Reformulate the helicopter model so that it has two inputs, the torque of the top rotor and the torque of the tail rotor.
- Show (by simulation) that if the top rotor applies a constant torque, then our controller cannot keep the helicopter from rotating. Increasing the feedback gain, however, reduces the rate of rotation.
- A better controller would include an integrator in the controller. Such controllers are studied in control systems theory.

Questions

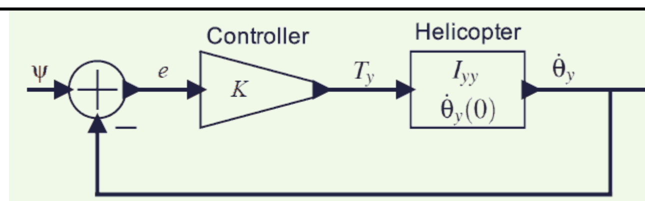
- Can the behavior of this controller change when it is implemented in software?
- How do we measure the angular velocity in practice? How do we incorporate noise into this model?
- What happens when you have failures (sensors, actuators, software, computers, or networks) https://www.youtube.com/watch?v=MhEXXgiIVuY

Next Lecture

- Architecture design
  - Block diagrams
  - Sequence diagrams

SPARE SLIDES

Behavior of the controller



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

Assume that helicopter is initially at rest,

$$\dot{\theta}(0) = 0,$$

and that the desired signal is

$$\psi(t) = au(t)$$

for some constant a.
By calculus (see notes), the solution is

$$\dot{\theta}_y(t) = au(t)(1 - e^{-Kt/I_{yy}})$$