Embedded and Cyber-Physical Systems

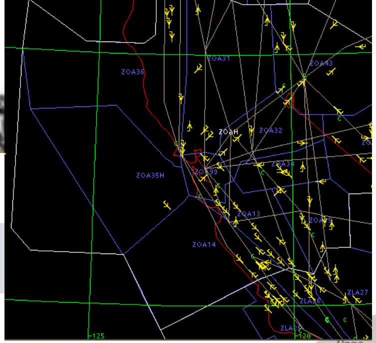
- Introduction -

What software can do

Cyber-Physical Systems (CPS):

Orchestrating networked computational resources with physical systems

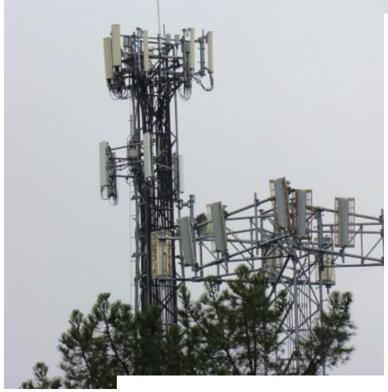
Avionics

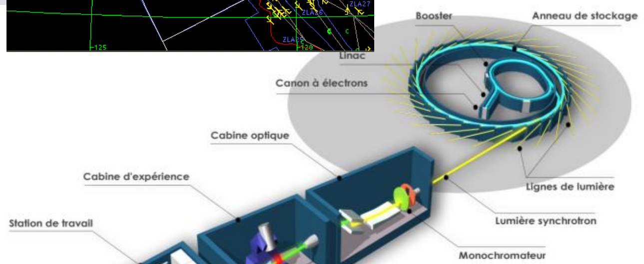


Transportation (Air traffic control at SFO)

Building Systems

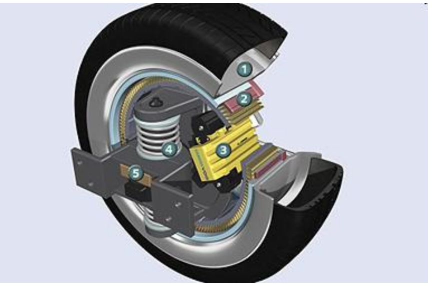
Telecommunications



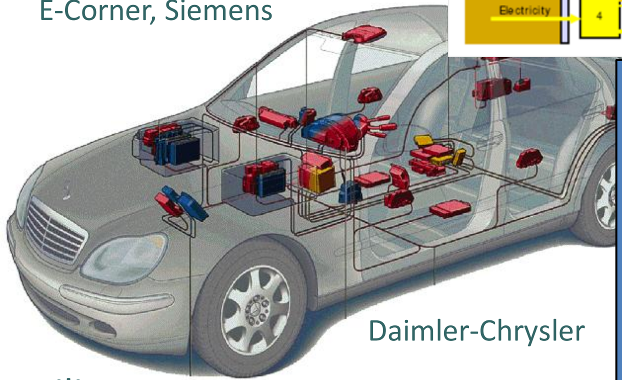


Instrumentation (Soleil Synchrotron)

Automotive



E-Corner, Siemens



Daimler-Chrysler

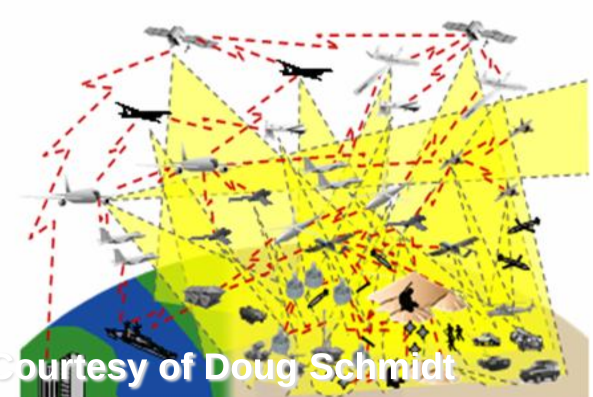
Power generation and distribution

Courtesy of General Electric

Factory automation

Courtesy of Kuka Robotics Corp.

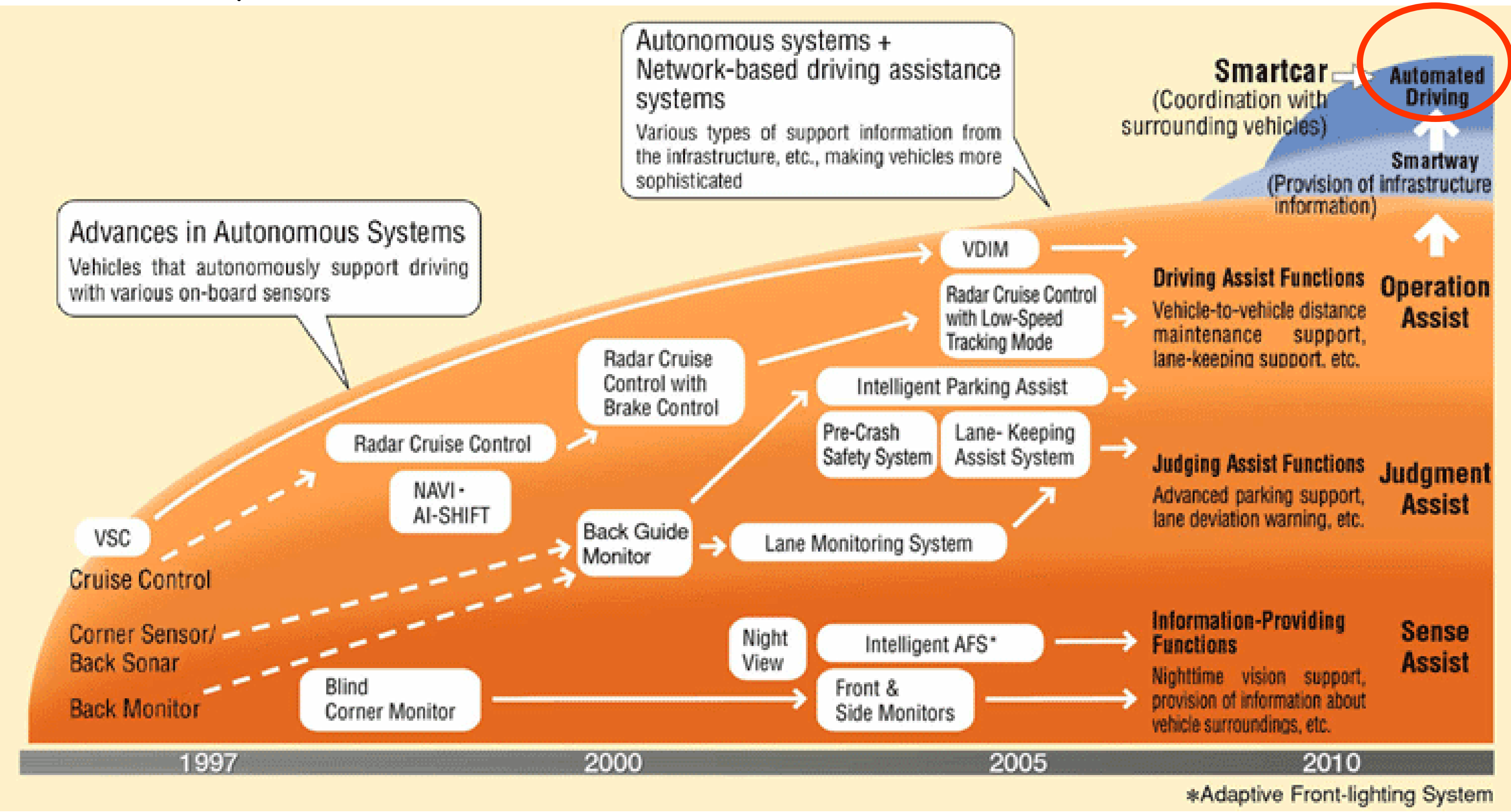
Military systems:



Slide from Lee & Seshia

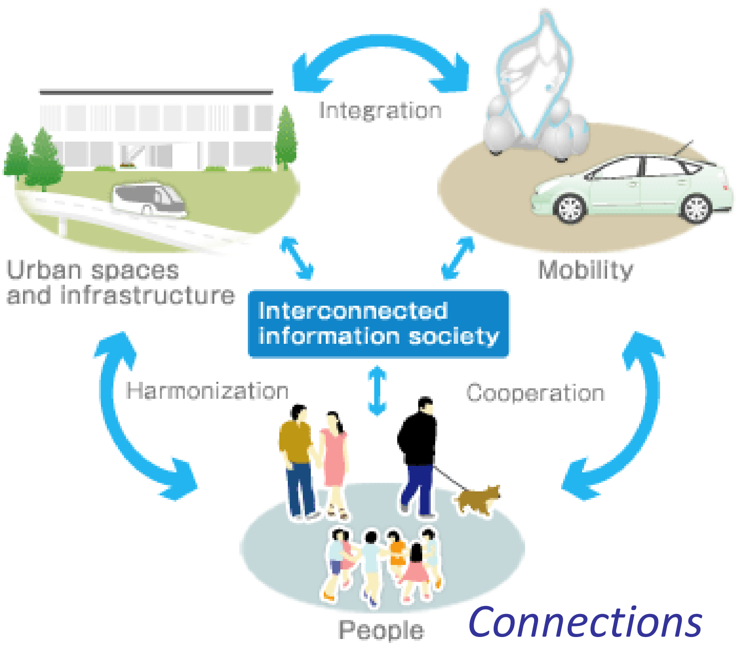
Example: Toyota autonomous vehicle technology roadmap, c. 2007

Source: Toyota Web site

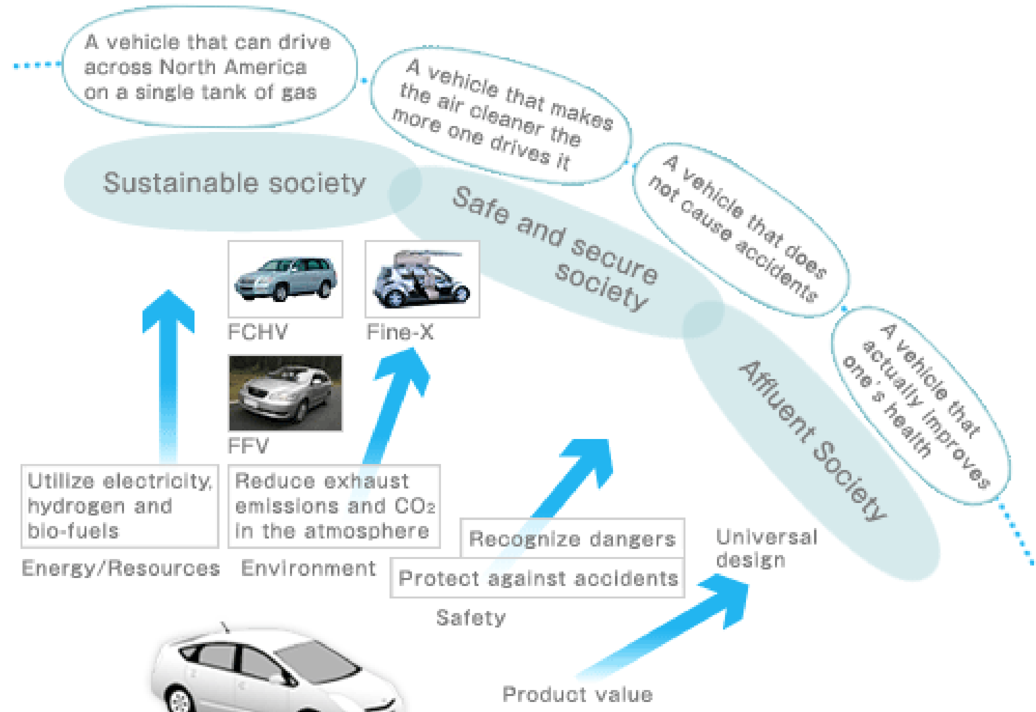


Toyota's Direction

Our sustainable mobility strategy includes products, partnerships, the urban environment and energy solutions.*



Connections



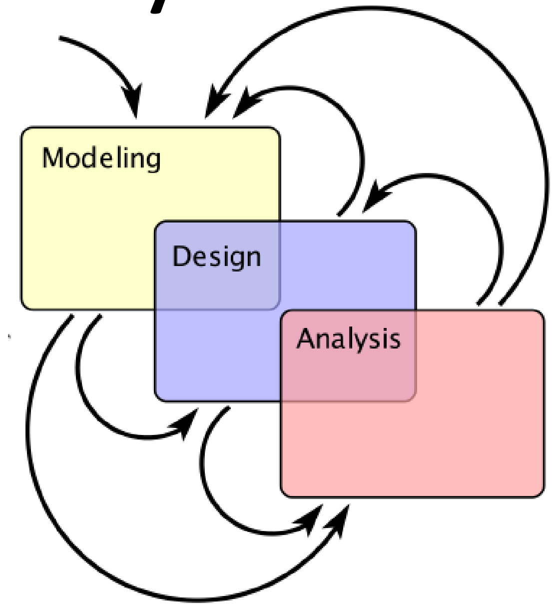
Vehicle technologies

Infrastructure integration

*Toyota 2010 North American Environmental Report Highlights

How can we deal with such complex systems?

Modeling, Design, Analysis



- ***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).

What is Modeling?

- Developing insight about a system, process, or artifact through imitation.

- A *model* is the artifact that imitates the system, process, or artifact of interest.

- A *mathematical model* is model in the form of a set of definitions and mathematical formulas.

What is Model-Based Design?

1. Create a model of all the parts of the embedded system
   – Physical world
   – Control system
   – Software environment
   – Hardware platform
   – Network
   – Sensors and actuators
2. Construct the implementation from the model
   – Construction may be automated, like a compiler
   – More commonly, portions are automatically constructed

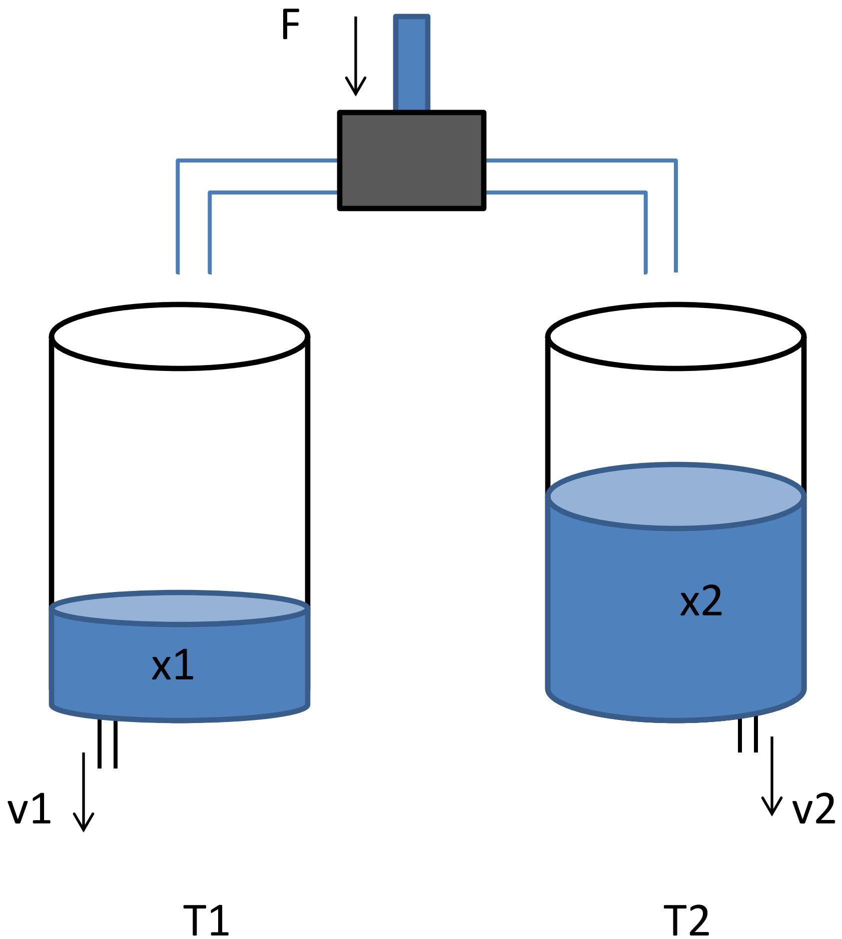
Modeling Techniques

- Models that are abstractions of **system dynamics** (how things change over time)

- Examples:
  - Modeling physical phenomena – ODEs
  - Feedback control systems – time-domain modeling
  - Modeling modal behavior – FSMs, hybrid automata
  - Modeling sensors and actuators – calibration, noise
  - Modeling software – concurrency, real-time models
  - Modeling networks – latencies, error rates, packet loss

Modeling of continuous dynamics

The two-tank example revisited



A mathematical model

$$(1)\begin{cases}\dot{x}_1 = F - v_1 \\ \dot{x}_2 = -v_2\end{cases}$$

$$(2)\begin{cases}\dot{x}_1 = -v_1 \\ \dot{x}_2 = F - v_2\end{cases}$$

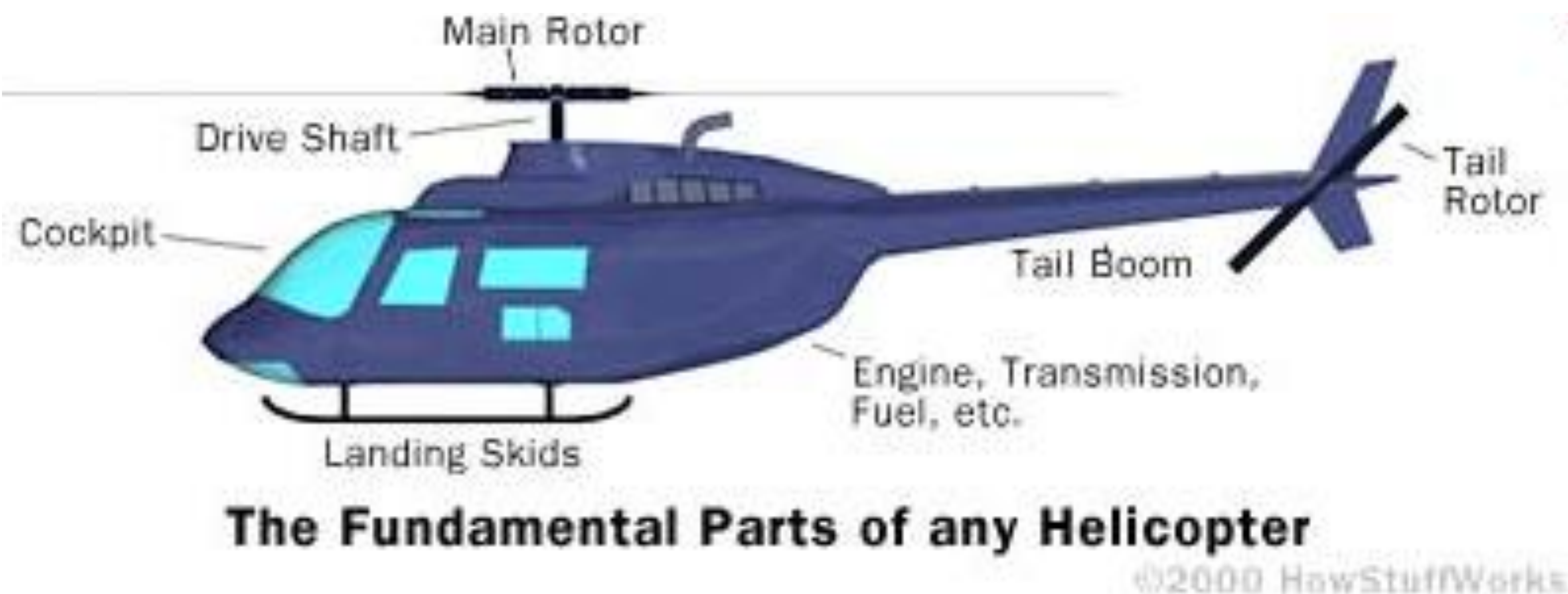
Solution

$$\begin{cases} x_1(t) = x_1^0 + (F - v_1)\cdot \Delta t_1 + (-v_1)\cdot \Delta t_2 \\ x_2(t) = x_2^0 + (-v_2)\cdot \Delta t_1 + (F - v_2)\cdot \Delta t_2 \end{cases}$$

Does a viable controller exist?
Check this:

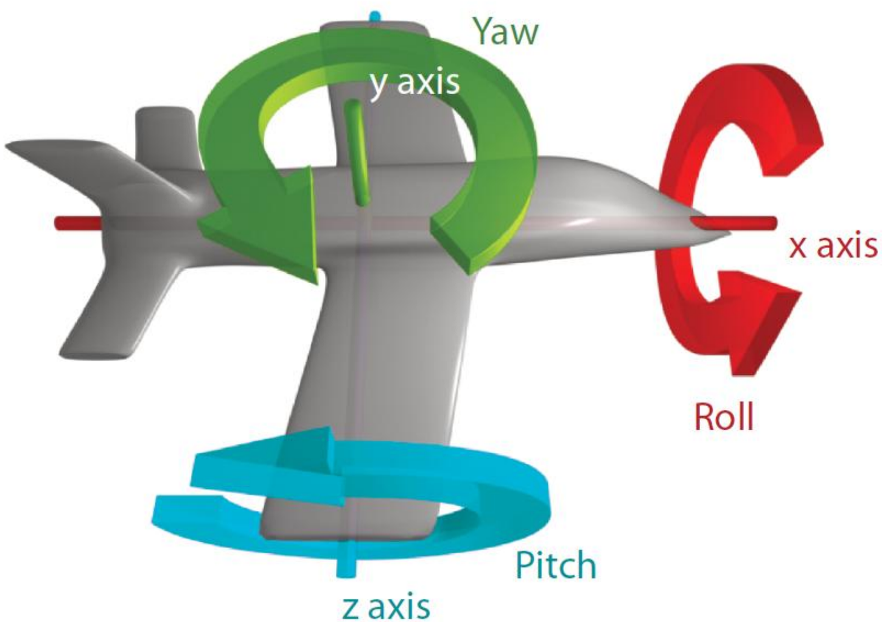
$$\begin{cases} (F - v_1)\cdot \Delta t_1 + (-v_1)\cdot \Delta t_2 \geq 0 \\ (-v_2)\cdot \Delta t_1 + (F - v_2)\cdot \Delta t_2 \geq 0 \end{cases}$$

An Example: Modeling Helicopter Dynamics



Modeling Physical Motion

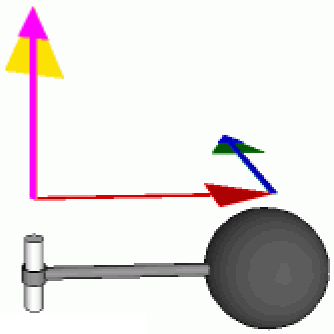
- Six degrees of freedom:
  - Position: x, y, z
  - Orientation: pitch, yaw, roll



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

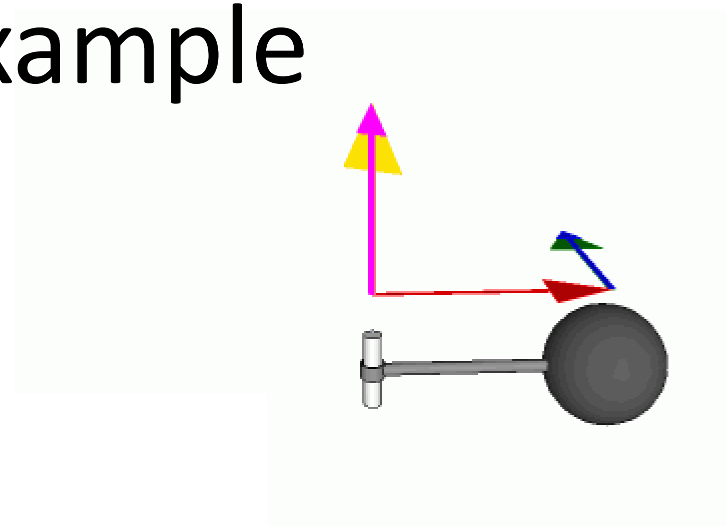
$$\mathbf{T}(t) = \frac{d}{dt}\left(I(t)\dot{\theta}(t)\right),$$

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.

Simple Example



Yaw dynamics:

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

To account for initial angular velocity, write as

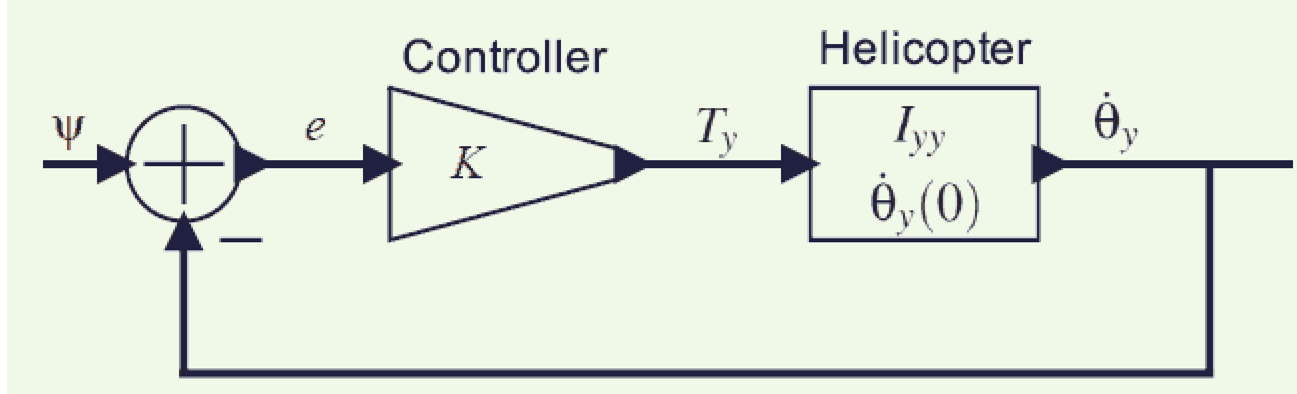
$$\dot{\theta}_y(t) = \dot{\theta}_y(0) + \frac{1}{I_{yy}}\int_0^t T_y(\tau)d\tau.$$

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.

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

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