# CDS 101: Lecture 10.1
# Implementation of Feedback Systems

**Richard M. Murray**
**29 November 2004**

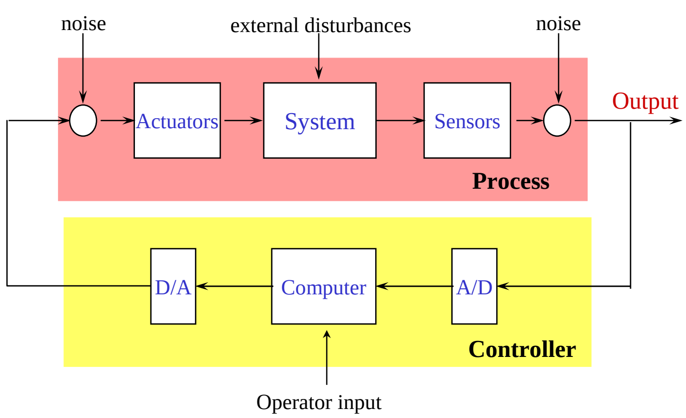
**Goals:**

- Describe how control laws are implemented in engineering systems
- Briefly summarize the main principles and tools for the course

**Reading:**

- Åström and Murray, *Analysis and Design of Feedback Systems,* Ch 10
- Optional: R. M. Murray (ed), *Control in an Information Rich World,* Available online at http://www.cds.caltech.edu/~murray/cdspanel or outside 109 STL

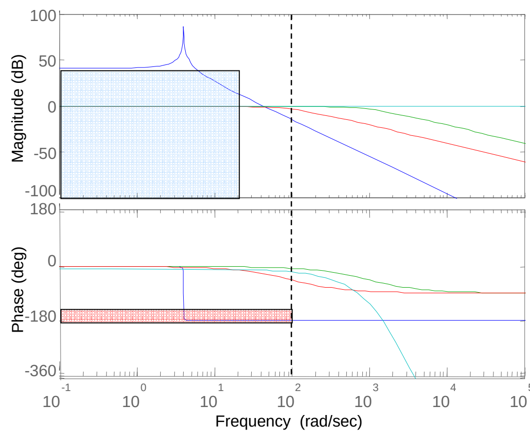
## Modern Control System Components




| | |
|---|---|
| **Process** | Physical system, actuation, sensing |
| **Controller** | Microprocessor plus conversion hardware (single chip) |
| **Feedback** | Interconnection between plant output, controller input |

29 Nov 04 R. M. Murray, Caltech CDS 2

## Frequency Response of Control Components




Plant: second order, tracking to 20 rad/sec

Sensing: 1000 rad/s bandwidth (1st order lag)

Actuation: 100 rad/s bandwidth (1st order lag)

Computation: 500 Hz (3000 rad/sec) update

**Components limit performance**

- Often ignore sensor, actuator, computation dynamics when designing controllers
- Each adds dynamics that limit achievable bandwidth
- OK to ignore if components are sufficiently high performance

**General guidelines**

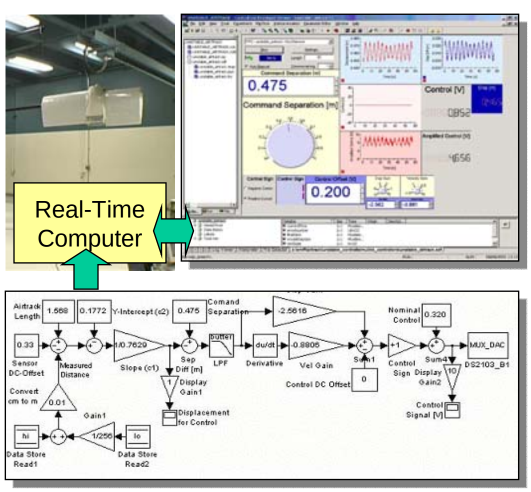
- Sensor: 5-10X BW
- Actuation: 2-5X BW
- Computing: 10-20X BW

**Other issues**

- Sensor aliasing – filter response should be small at ½ sampling rate

29 Nov 04 R. M. Murray, Caltech CDS 3

## Modern Implementation Environments




**Graphical control environments**

- Define controller using traditional block dia-grams
- Program generates C code corresponding to digital implementation
  - Create state space realization of transfer functions
  - Convert to discrete time equivalent
- Compile code on target processor environment (PC or embedded proc)
- Real-time interface allows monitoring and debugging

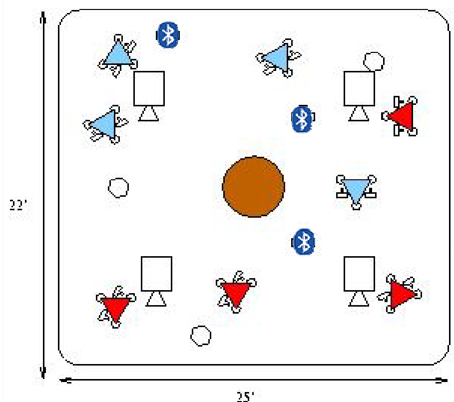
**Example: dSPace real-time control system**

- Compiles Simulink diagrams for DSP processors

29 Nov 04 R. M. Murray, Caltech CDS 4

Cremean et al
CDC 2002

## Example: Multi-Vehicle Wireless Testbed (MVWT)



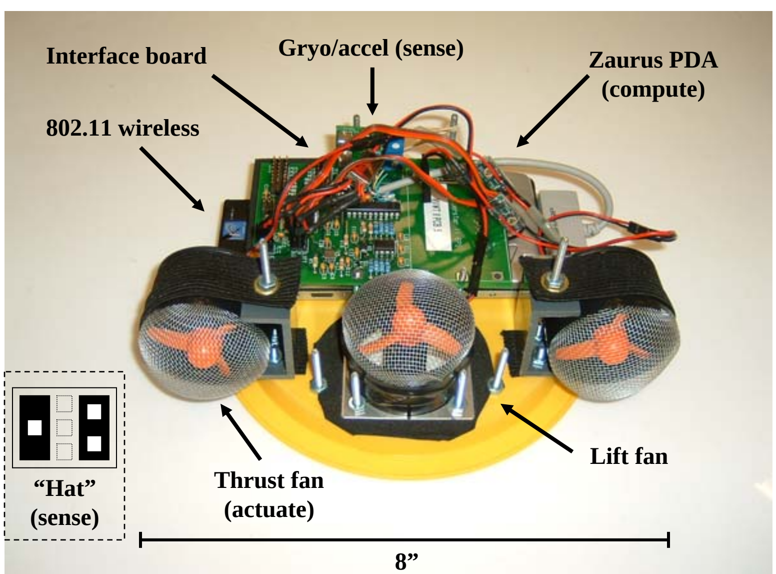

**Testbed features**

- Distributed computation on vehicles + command and control console
- Point to point networking (bluetooth) + local area networking (802.11)
- Overhead vision system provides global position data (LPS)

29 Nov 04 R. M. Murray, Caltech CDS 5

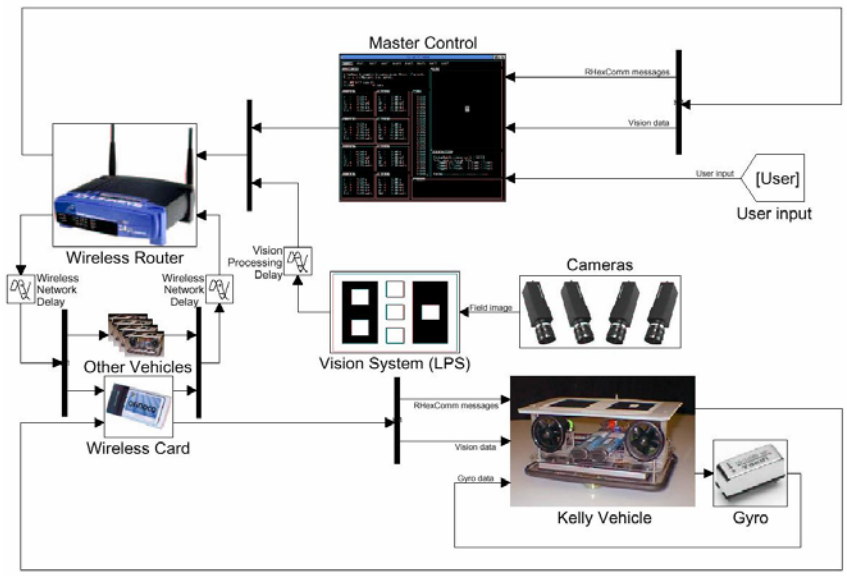
Jin, Waydo, Wildanger, Lammers, Scholze, Foley, Held, Murray
ACC 2004

## Caltech Hovercraft



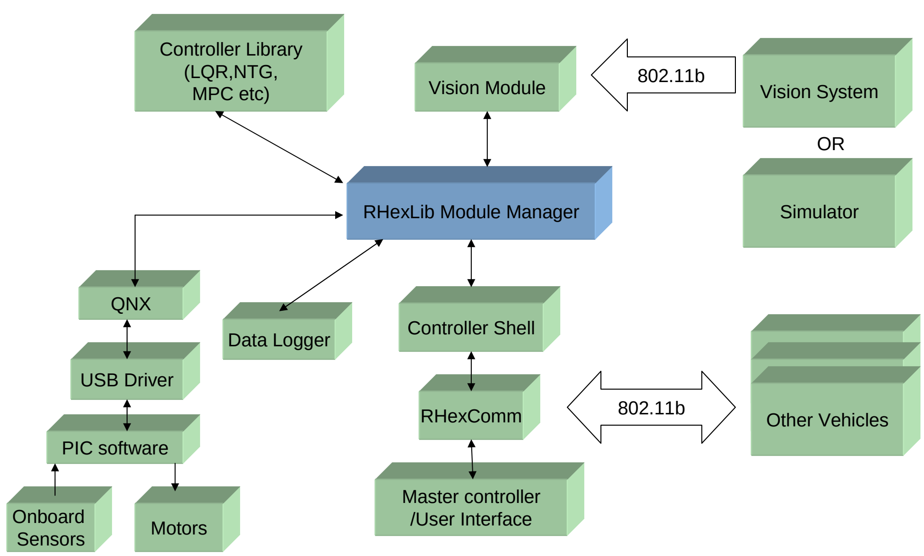

29 Nov 04 R. M. Murray, Caltech CDS 6

## MVWT Block diagram



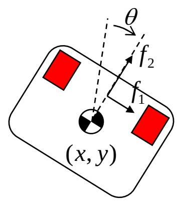

29 Nov 04 R. M. Murray, Caltech CDS 7

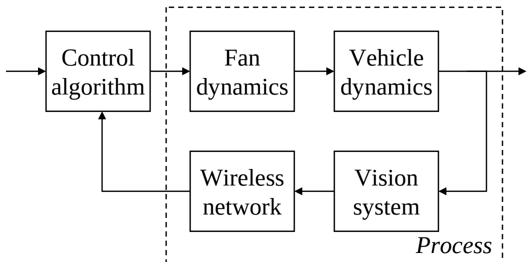
## Software Architecture




29 Nov 04 R. M. Murray, Caltech CDS 8

## Dynamics




**Vehicle dynamics:**

$$m\ddot{x} = f_1 \cos\theta - f_2 \sin\theta - d\dot{x}$$
$$m\ddot{y} = f_1 \sin\theta + f_2 \cos\theta - d\dot{y}$$
$$J\ddot{\theta} = r f_1$$

**Fan dynamics**

$$m\dot{\omega}_1 = -\tau\omega_1 + \alpha(i_1) \qquad f_1 = N\omega_1$$
$$m\dot{\omega}_2 = -\tau\omega_2 + \alpha(i_2) \qquad f_2 = N\omega_2$$

Nonlinear map

**Vision System**

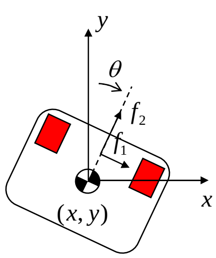
- Time delay + noise (pixelation)

**Wireless network**

- Variable time delay

29 Nov 04 R. M. Murray, Caltech CDS 9

## Linearization




$$m\dot{v} = f_1 \cos\theta - f_2 \sin\theta - dv$$
$$m\ddot{x} = f_1 \sin\theta + f_2 \cos\theta - d\dot{x}$$
$$J\ddot{\theta} = r f_1$$

Note: omitted *y* from state

**Linearize around a constant velocity along *y***

$$v = v_0 \qquad f_1 = dv_0$$
$$x = 0, \dot{x} = 0 \qquad f_2 = 0$$
$$\theta = 0, \dot{\theta} = 0$$

**Shift coordinates to the origin and write in state space form:**

$$\tilde{v} = v - v_0$$
$$u_1 = f_2$$
$$u_2 = f_1 - dv_0$$

$$\frac{d}{dt}\begin{bmatrix} \tilde{v} \\ y \\ \theta \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} -d/m & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -dv_0/m & -d/m & 0 \\ 0 & 0 & 0 & 0 & -b/J \end{bmatrix} \begin{bmatrix} \tilde{v} \\ y \\ \theta \\ \dot{y} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 1/m & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1/m \\ 0 & -r/J \end{bmatrix} u$$

**Remarks**

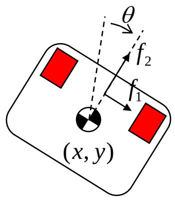
- Ignores actuator dynamics (assume fast)
- Ignores time delays (for now)

**Parameters (Kelly II)**

m = 5.5 b = 0.1
J = 0.047 r = 0.123
d = 0.5

29 Nov 04 R. M. Murray, Caltech CDS 10

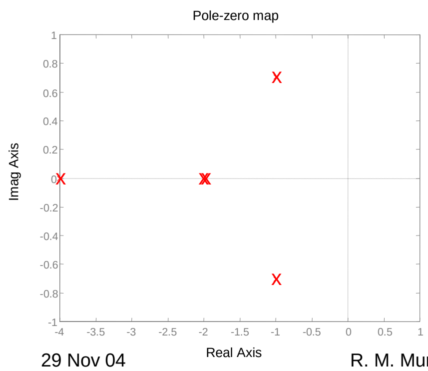
## Control Design: State Space

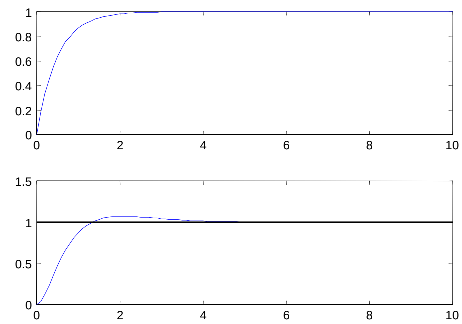



$$\frac{d}{dt}\begin{bmatrix}\tilde v\\ y\\ \theta\\ \dot y\\ \dot\theta\end{bmatrix} = \begin{bmatrix}-d/m & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 1\\ 0 & 0 & -dv_0/m & -d/m & 0\\ 0 & 0 & 0 & 0 & -b/J\end{bmatrix}\begin{bmatrix}\tilde v\\ y\\ \theta\\ \dot y\\ \dot\theta\end{bmatrix} + \begin{bmatrix}1/m & 0\\ 0 & 0\\ 0 & 0\\ 0 & 1/m\\ 0 & -r/J\end{bmatrix} u$$

**Choose control to stabilize error, $e = x - x_d \rightarrow u = K(x-x_d)+u_d$**

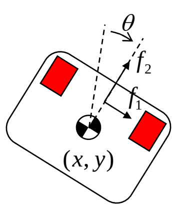
- $x_d$ = desired state
- $u_d$ = nominal force





## Control Design: Frequency Domain

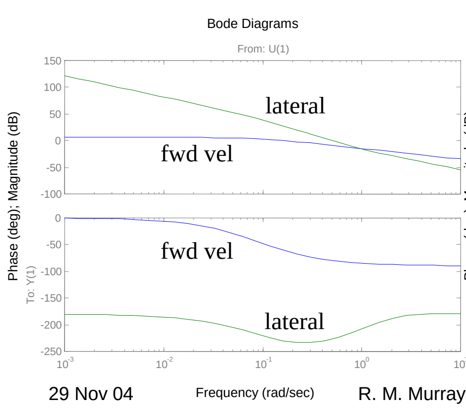


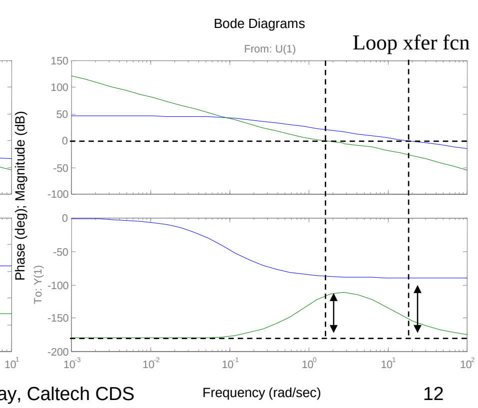

$$P_{\tilde v u_1} = \frac{1}{s+d} \qquad P_{yu_2} = \frac{Js^2 + bs + Jdv_0/m}{(ms^2+ds)(Js^2+bs)}$$

**Compute transfer functions using $H = C(sI-A)^{-1}B$**

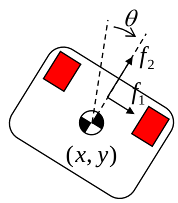
**Use loop shaping to design compensator**

- Forward velocity: simple proportional gain
- Lateral position: use lead compensator



## Control Design: Frequency Domain




$$P_{\tilde{v}u_1} = \frac{1}{s+d}$$

$$C_v = K_p$$

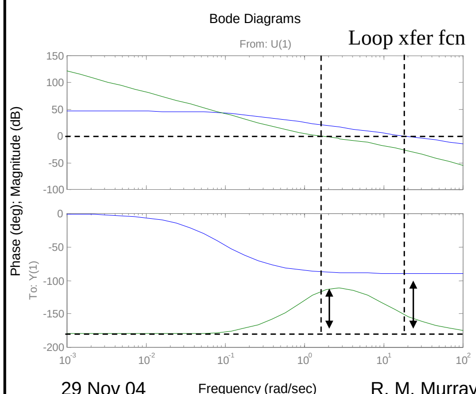
Kp = 100

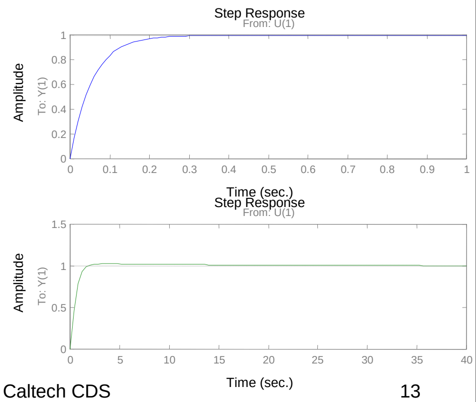
$$P_{yu_2} = \frac{Js^2 + bs + Jdv_0/m}{(ms^2 + ds)(Js^2 + bs)}$$

$$C_l = K\frac{s+a}{s+b}$$

a = 0.1 K = 100

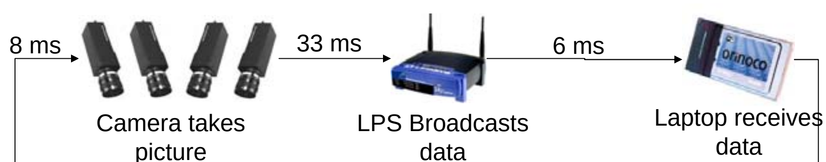
b = 10



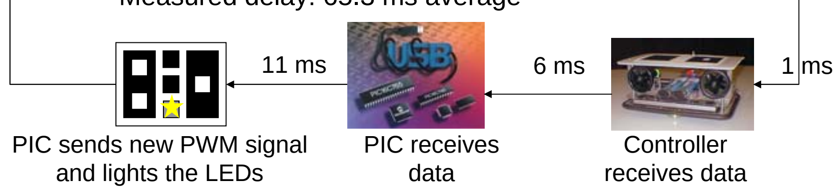

29 Nov 04 R. M. Murray, Caltech CDS 13

## Time Delay Estimates



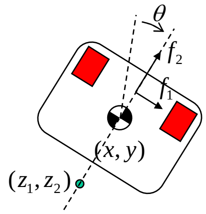


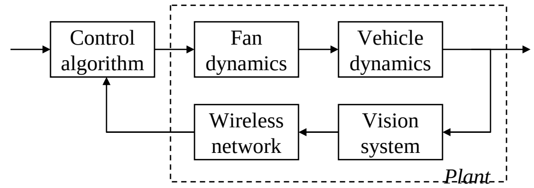
- No actuator delay included in estimates or measurement
- Total delay estimate: 65 ms
- Measured delay: 65.3 ms average

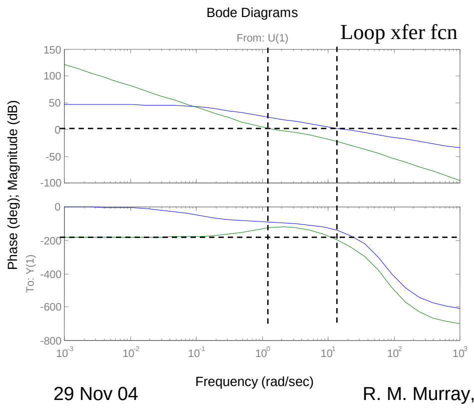


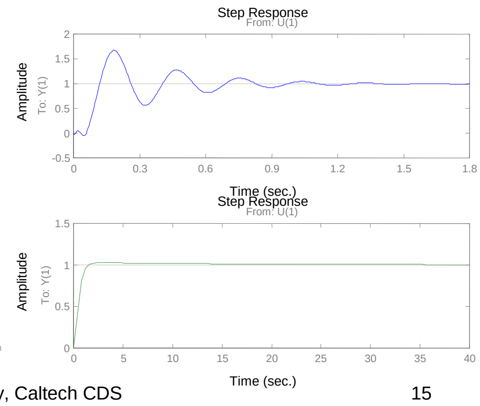

29 Nov 04 R. M. Murray, Caltech CDS 14

## Unmodeled Dynamics: Sensor Delay



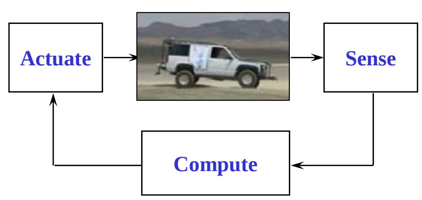




29 Nov 04 R. M. Murray, Caltech CDS 15

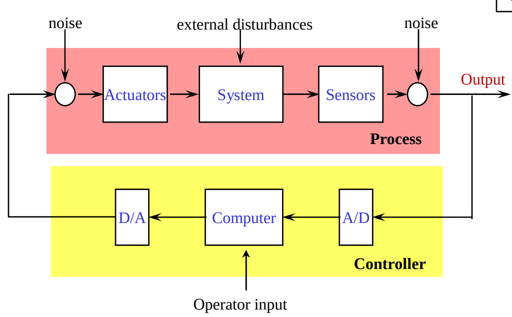
## Summary: Control Implementation

**Control = sense, compute, actuate**

- Computation can be done in analog or digital form
- Sensing should be high bandwidth, low noise for best performance
- Modern environments allow direct implementation through Simulink




Stability Performance Robustness




29 Nov 04 R. M. Murray, Caltech CDS 17