Making Robots and Making Robots *Intelligent*

Professor Jorge Cortes
Applied Math and Statistics

Professor Bill Dunbar
Computer Engineering

Instructor William Thompson
Transferable Skills

TODAY

- Introduction to Robotics
- Intelligence through feedback control
- Success requires modeling

  ==> which requires ***math***
- You will be *Intelligence Engineers!*
- Overview of the next 4 weeks
- What we won't cover

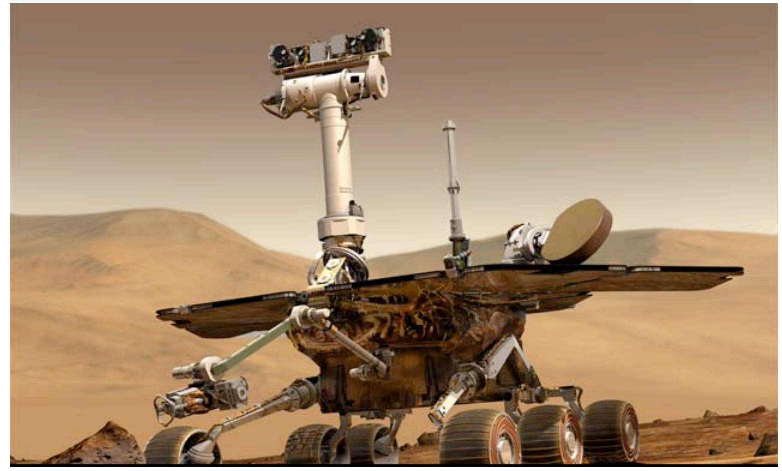
Robots

According to Merriam-Webster:

1 : A machine that looks like a human being and performs various complex acts (as walking or talking) of a human being; *also* **:** a similar but fictional machine whose lack of capacity for human emotions is often emphasized.

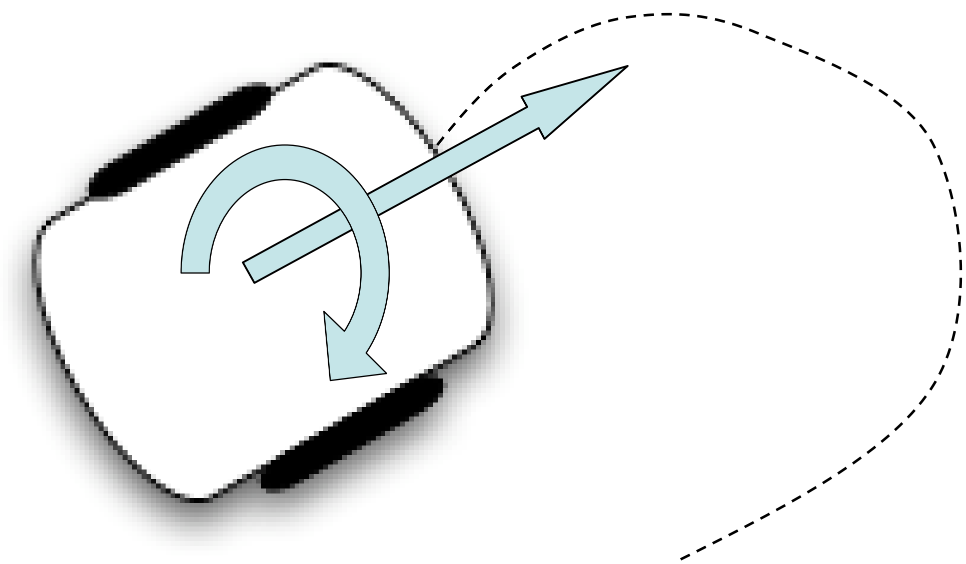
2 : A device that automatically performs complicated often repetitive tasks.

3 : A mechanism guided by automatic controls



An Example: *Robobrain*

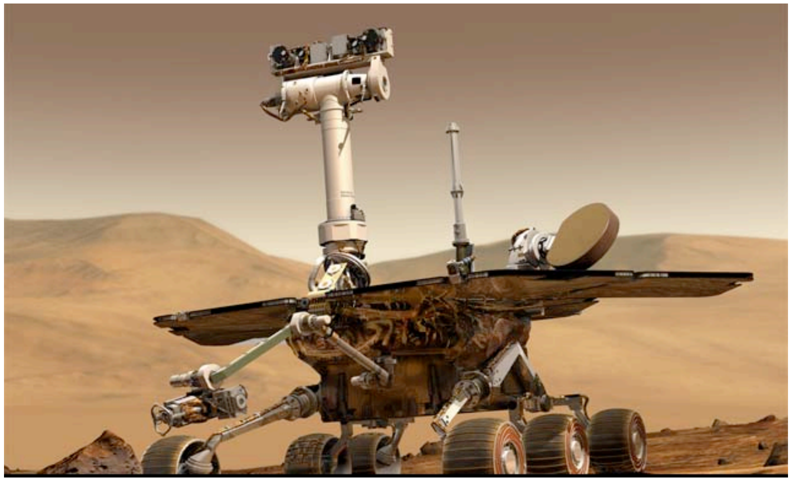
- A cart with two main wheels driven by motors
- Wheels independently driven ==> cart can turn or go straight
- Can measure distance to certain objects and detect lines



Objective: ***Autonomously*** track a line or follow a wall. ==> Intelligence (action, not comprehension or communication) accomplished by *guidance*, *navigation and (feedback) control - GNC.*

Guidance - How to Get There

- Finding a feasible path from where you are to where you want to go.
- Feasible means avoiding obstacles.
- State-of-the-art (in Space):



"Spirit has had an average daily driving distance of about 121 feet (37 meters) when assigned to long treks. Mission planners would like to see it boosted up to at least 164 feet (50 meters) a day so that the rover can eventually reach some hills that lie a month's travel or more in the distance."

Source: http://www.space.com/

Key Issue: Guidance paths assume ideal conditions

Navigation - Where I Am

- How do you know where you are?
  - Outdoors
  - Underwater
  - In Space
  - Indoors
  - Underground

- *Sensors (GPS, acoustics, IR, LADAR, memory, etc.)*
- Robobrain - IR sensors detect distance to objects

What is Feedback?

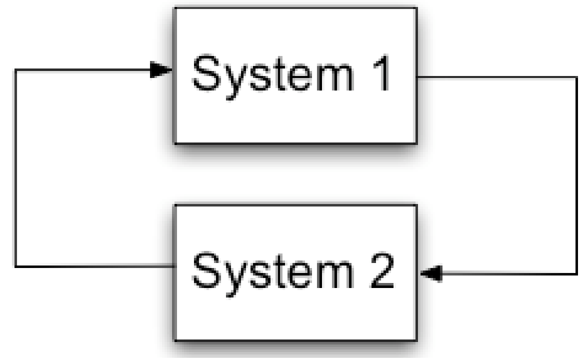
Merriam-Webster:

the return to the input of a part of the output of a machine, system, or process (as for producing changes in an electronic circuit that improve performance or in an automatic control device that provide self-corrective action) [1920]

Feedback = presence of interconnection of two (or more) systems

- System 1 affects System 2
- System 2 affects System 1
- Cause and effect is tricky, as

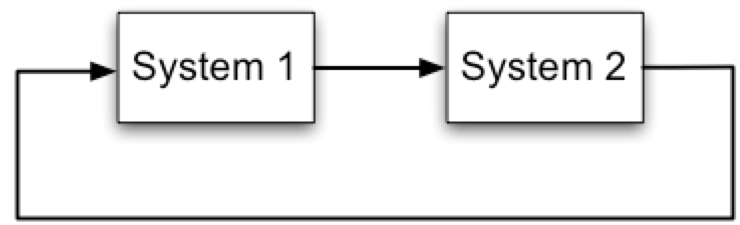
systems are ***mutually dependent***



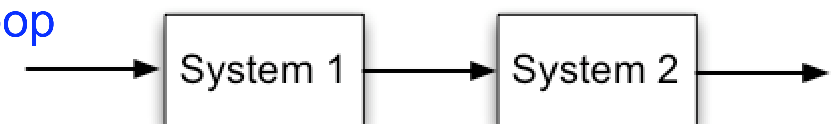
(e.g. Predator/Prey Populations)

Terminology

Closed-loop



Open-loop



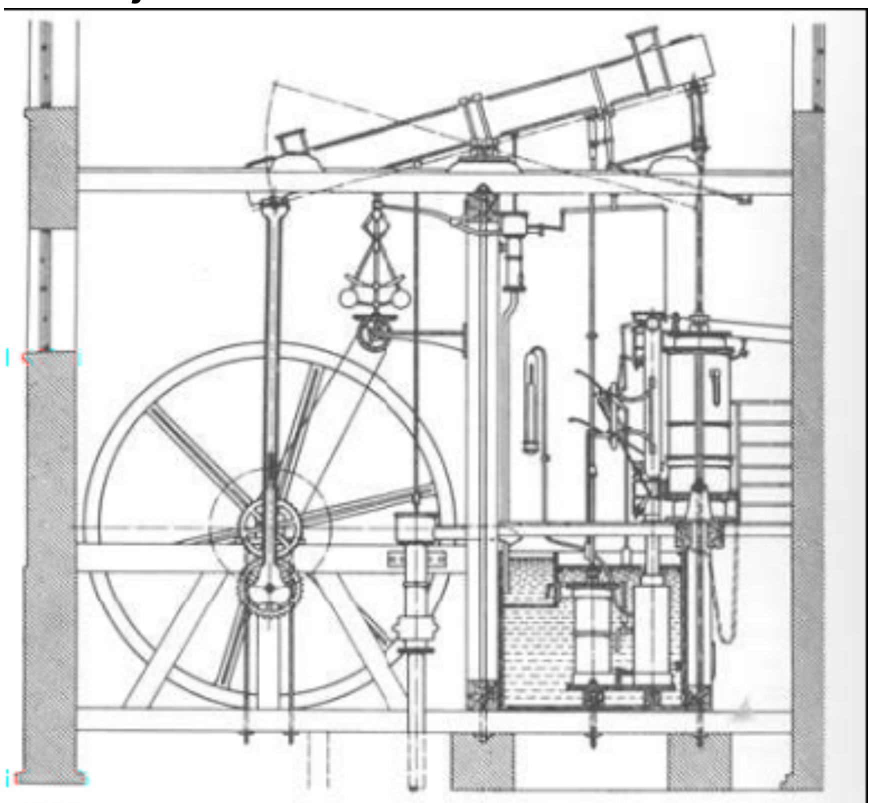
(e.g. Predator/Predator Populations)

Feedback is ubiquitous in natural and engineered systems

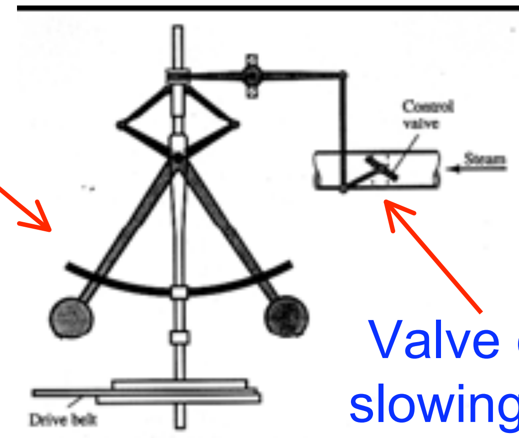
Closed-loop Example: Flyball Governor

"Flyball" Governor (1788)

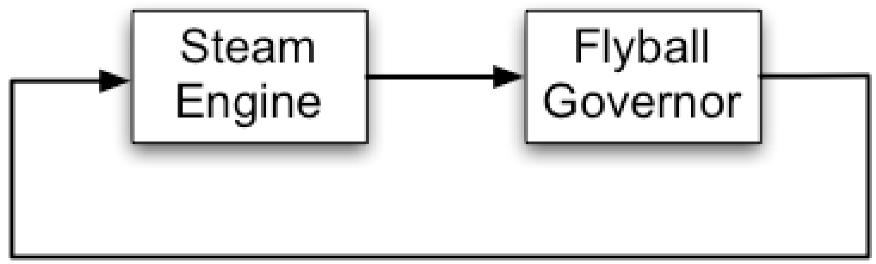
- Regulate speed of steam engine
- Reduce effects of variations in load (disturbance rejection)
- Major advance of industrial revolution



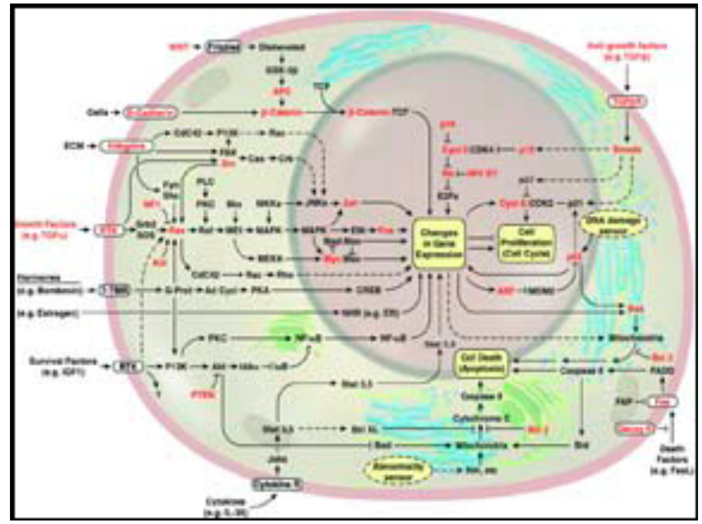
Balls fly out as speed increases



Valve closes, slowing engine



Other Examples of Feedback



Biological Systems

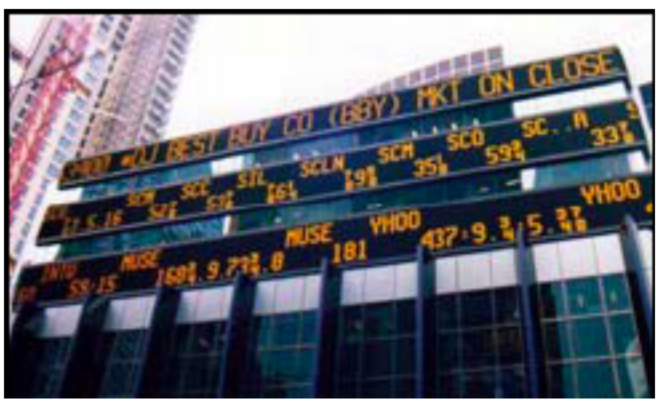
- Physiological regulation (homeostasis)
- Biomolecular regulatory networks

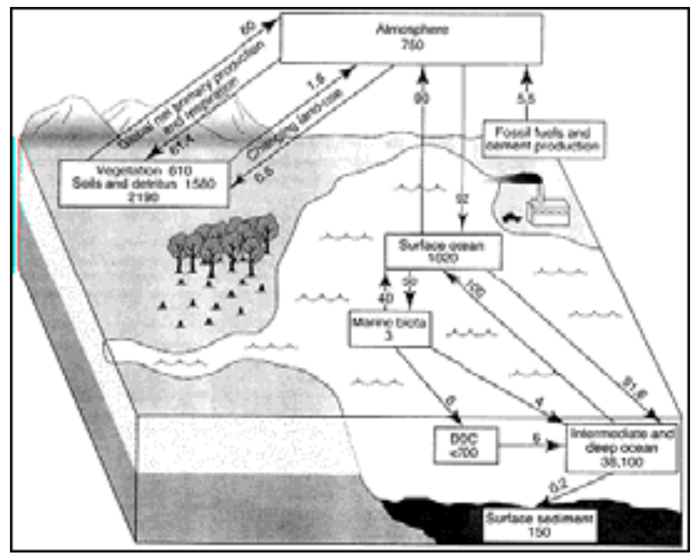
Environmental Systems

- Microbial Ecosystems
- Global Carbon Cycle

Financial Systems

- Markets and Exchanges
- Supply and Service Chains





GNC: Feedback *Control* Realizes the Guided Behavior in the Presence of Uncertainty

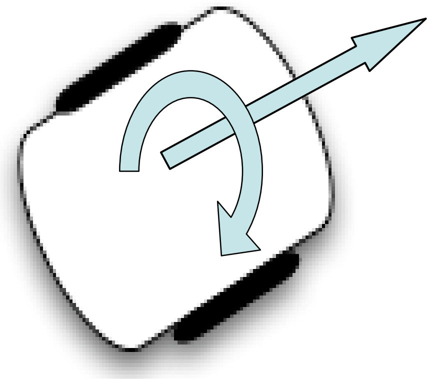
Goals

- Stability: system maintains desired operating point (hold steady speed)
- Performance: achieve desired *ASAP*
- Robustness: system tolerates perturbations in dynamics (mass, drag, etc)

Two Main Principles of Control

Robustness to Uncertainty through Feedback

• Feedback allows high performance in the presence of uncertainty
• Example: repeatable performance of robobrain with non-matching motors/wheels
• Key idea: *sensing* to compare actual to desired & correction through *computation* and *actuation*



Shaping Dynamic Behavior through Feedback

• Feedback allows the dynamics of a system to be modified
• Example: stability augmentation for highly agile, unstable aircraft
• Key idea: interconnection gives *closed-loop* that modifies natural behavior

X-29 experimental aircraft

Example: Cruise Control - Model comes First

Update Rule for Model (in words):

New velocity (v_{k+1}) is the old velocity (v_k) plus the effects of control [engine] (u_{eng}) and disturbances [drag/friction & hill] ($-bv_k$, u_{hill}).

Parameters:

m: mass, b: drag/friction coefficient

Δ: time between updates

Model of "*Bob*" describes updates in speed every Δ seconds:

$$mv_{k+1} = mv_k + \Delta[-bv_k + u_{eng} + u_{hill}]$$

Example: Cruise Control

Model of "*Bob*":

$$mv_{k+1} = mv_k + \Delta[-bv_k + u_{eng} + u_{hill}]$$

Control:

$$u_{eng} = K(v_{des} - v_k), \quad K > 0$$

Steady-state (when $v_k = v_{ss}, \;\; k = 0, 1, 2, \ldots$):

$$\Longrightarrow v_{ss} = \underbrace{\frac{K}{b+K}}_{\text{Goes to 1 as } K \to \infty} v_{des} + \underbrace{\frac{1}{b+K}}_{\text{Goes to 0 as } K \to \infty} u_{hill}$$

Goes to 1 as $K \to \infty$

Goes to 0 as $K \to \infty$

Stability/performance: Steady state velocity approaches desired velocity as $K \to \infty$. No overshoot or oscillations.

Disturbance rejection: Effect of disturbances (hills) approaches zero as $K \to \infty$

Robustness: Results don't depend on the specific values of b, m, or *K* for *K* large

Objective: DO THIS WITH ROBOBRAIN!

Clearly, the **Model** of "*Bob*" played a key role in the analysis that demonstrated the effectiveness of the control***…and it only required Algebra!***

We'll also see that ***plotting response*** to gauge effectiveness is just as important.

In next four weeks, you will learn tools so that you can design and implement a control to get robobrain to follow a wall…but, it will only happen if you are willing to ***work and ask lots of questions.***

Overview of Next 4 Weeks

<u>Week 1:</u>

- Matlab
- Plotting functions
- Introduction to Modeling
- Cool example: Chaos! (are you up to it?)

Overview of Next 4 Weeks

<u>Week 2:</u>

- Modeling
- Robobrain Model
- Intro to Feedback
- Feedback Control Design

Overview of Next 4 Weeks

Week 3:

- Design a Cruise Control
- Testing your Control
- Design and Test Robobrain Wall Following Control
- Constructing your own Robobrain

Overview of Next 4 Weeks

<u>Week 4:</u>

- Continued Design and Testing Robobrain Control
- Hardware Implementation
- Presentations - You choose Nanotech or Intelligent Robotics.

Not Covered (Still Important)

- Artificial Intelligence
- Behavior-based approaches
- System Identification
- Mechatronic design
- Path planning
- C Programming

Slide Acknowledgements

- Prof. Richard Murray, Caltech, for CDS 101 lecture slides and web page infrastructure.
- Prof. Gabriel Elkaim, UCSC, for robotics slides.