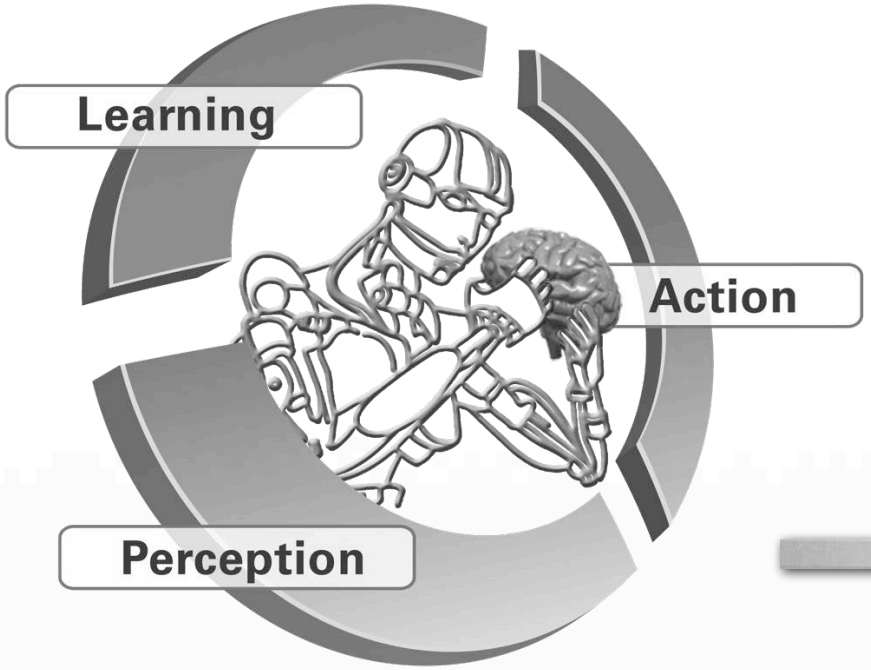
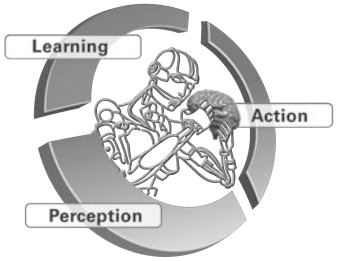


Robotics

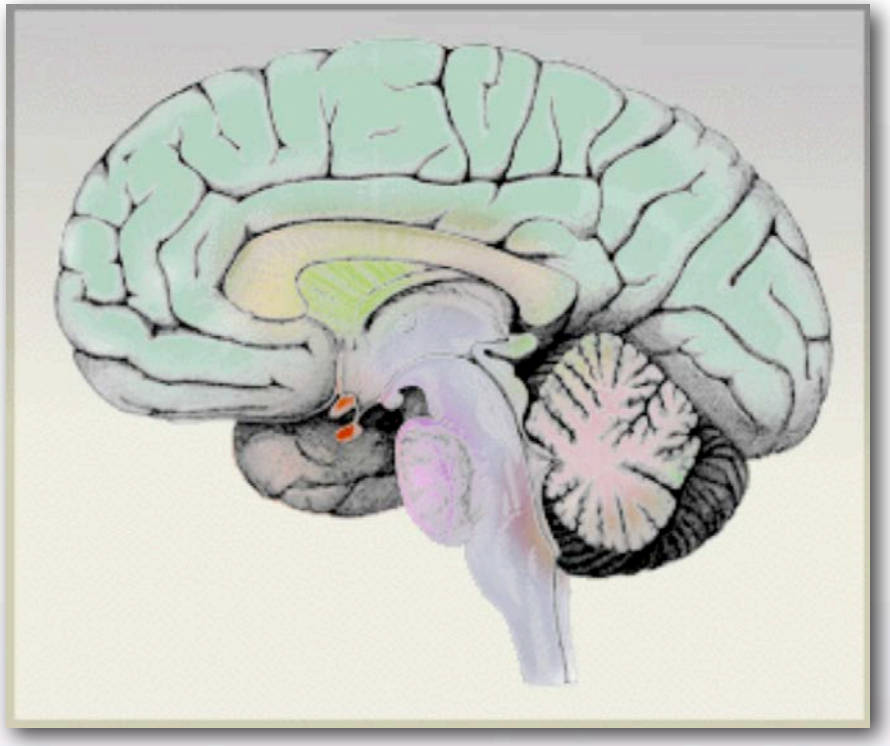
Part I: From Control to Learning
Model-based Control

Stefan Schaal
Max-Planck-Institute for Intelligent Systems
Tübingen, Germany
&
Computer Science, Neuroscience, & Biomedical Engineering
University of Southern California, Los Angeles

sschaal@is.mpg.de
http://www-amd.is.tuebingen.mpg.de

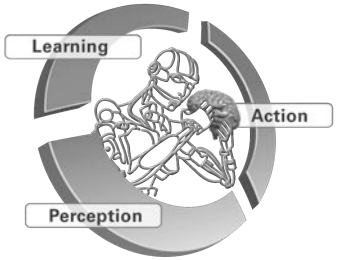


Grand Challenge #I: The Human Brain



How does the brain learn and control complex motor skills?

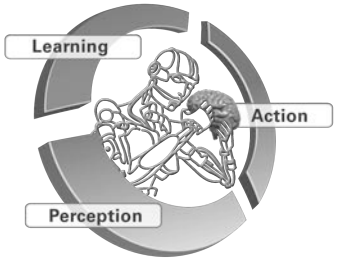
Applications: Facilitate learning, neuro-prosthetics, brain machine interfaces, movement rehabilitation, etc.



Example Application: Revolutionary Prosthetics

A Key Question for Research:

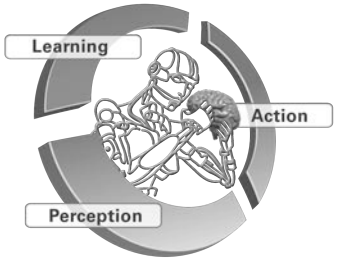
How does the brain control muscles and coordinate movement?



Grand Challenge #II: Humanoids

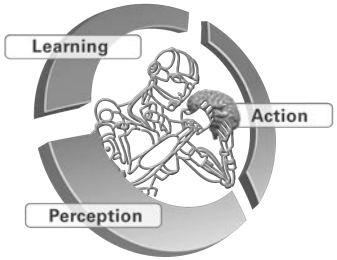
Can we create an autonomous humanoid robot?

Applications: assistive robotics, hazardous environments, space exploration, etc.

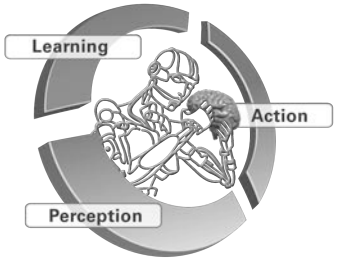


Example: The Hollywood View of Assistive Robotics

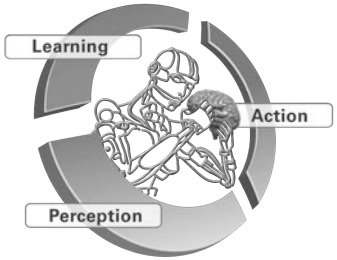
From the movie "I, Robot"



The Hollywood Future Is Not So Far ...

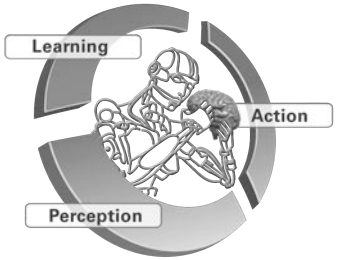


The Hollywood Future Is Not So Far ...



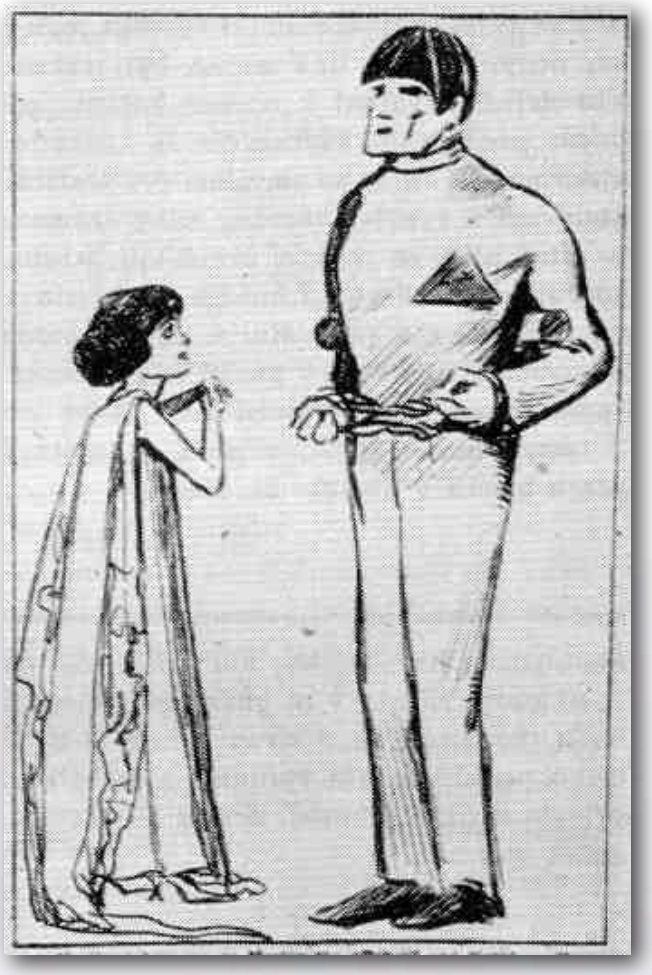
Outline

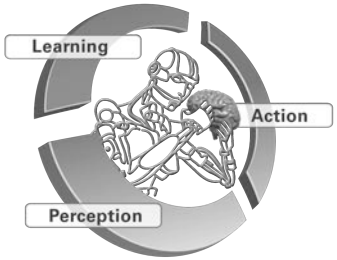
- A Bit of Robotics History
- Foundations of Control
- Adaptive Control
- Learning Control
  - Model-based Robot Learning



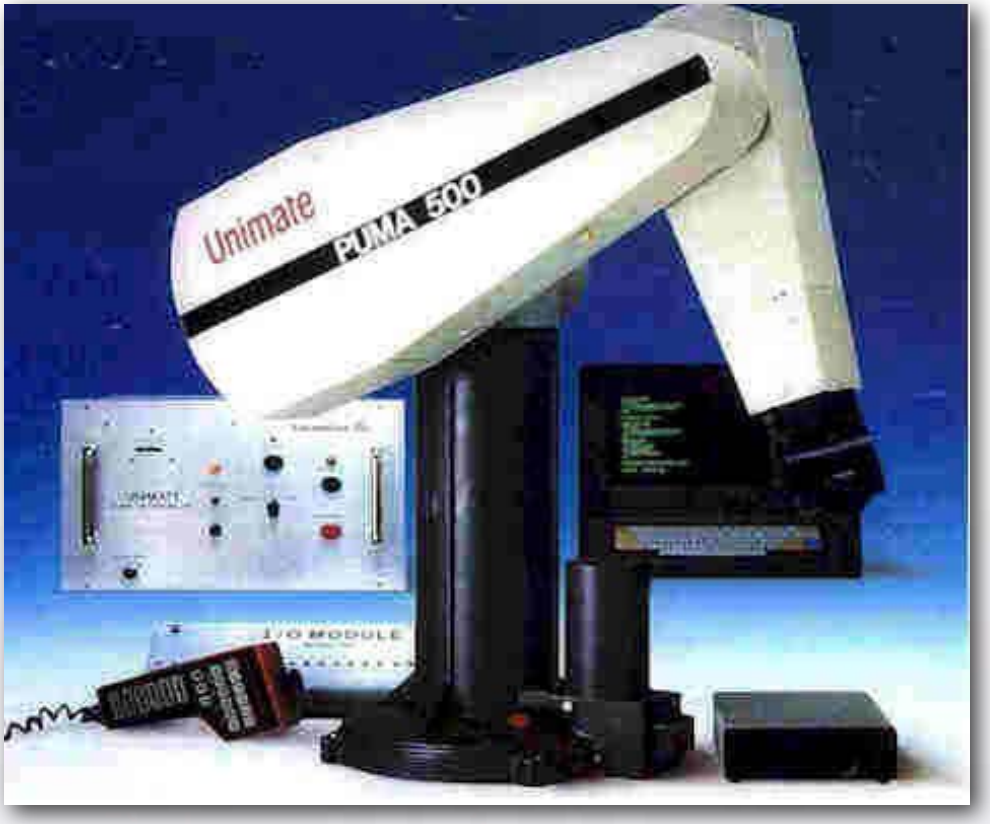
Robotics–The Original Vision

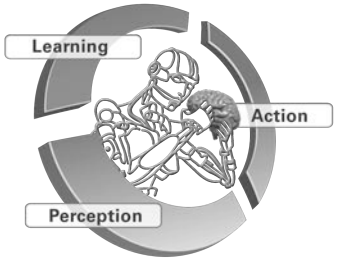
Karel Capek 1920:
Rossum's Universal Robots





Robotics–The Initial Reality





Some History Bullets

1750
Swiss craftsmen create automatons with clockwork mechanisms to play tunes and write letters.

1917
The word "robot" first appears in literature, coined in the play Opilek by playwright Karel Capek, who derived it from the Czech word "robotnik" meaning "slave."

1921
The term robot is made famous by Capek's play R.U.R. (Rossum's Universal Robots).

1938
Isaac Asimov coins the term robotics in his science fiction novels, and formulates the Three Laws of Robotics which prevent robots from harming humans.

1954
The first United Kingdom robotics patent, No. 781465, is granted in England on March 29.

1956
The Logic Theorist, an artificial intelligence machine capable of proving logical propositions point-by-point, is unveiled at Dartmouth College.

1958
Joseph F. Engelberger sets up a business in his garage called Consolidated Controls, making aircraft components. Joseph F. Engleberger and George C. Devol name their first robot "Unimate." The first Unimate is installed at a General Motors plant to work with heated die-casting machines. founds Unimation, the first commercial company to make robots. Unimation stood for Universal automation.

1960
Artificial intelligence teams at Stanford Research Institute in California and the University of Edinburgh in Scotland begin work on the development of machine vision.

1961
George C. Devol obtains the first U.S. robot patent, No. 2,998,237.

1961
First production version Unimate industrial robot is installed in a die-casting machine.

1961
The MH-1, Mechanical Hand with sensors, is developed at MIT by Ernst.

1962
Consolidated Diesel Electric Company (Condec) and Pullman Corporation enter into joint venture and form Unimation, Inc. (Unimation stood for "Universal Automation").

1963
The Versatran industrial robot became commercially available.

1964
The first Tralfa robot is used to paint wheelbarrows in a Norwegian factory during a human labor shortage.

1966
The first prototype painting robots are installed in factories in Byrne, Norway.

1966
The robotic spacecraft "Surveyor" (United States) lands on the moon.

1968
"Shakey," the first complete robot system is built at Stanford Research Institute, in California.

1968
Unimation takes its first multi-robot order from General Motors.

1969
Robot vision, for mobile robot guidance, is demonstrated at the Stanford Research Institute.

1969
Unimate robots assemble Chevrolet Vega automobile bodies for General Motors.

1970
General Motors becomes the first company to use machine vision in an industrial application. The Consight system is installed at a foundry in St. Catherines, Ontario, Canada.

1970
The Russian lunar rover Lunakhod, wheels about on the moon.

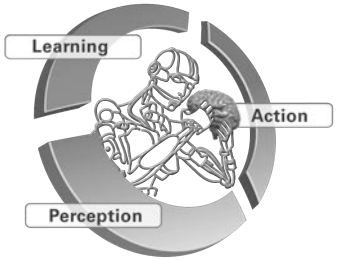
1970
The first American symposium on robots meets in Chicago.

1971
Japan establishes the Japanese Industrial Robot Association (JIRA), and becomes the first nation to have such an organization.

1972
The SIRCH machine, capable of recognizing and orienting randomly presented two-dimensional parts, is developed at the University of Nottingham, England.

1972
Kawasaki installs a robot assembly line at Nissan, Japan, using robots supplied by Unimation, Inc.

1973
"The Industrial Robot," the first international journal of robotics, begins publication.



Some History Bullets

1973
The ASEA Group of Vasteras, Sweden, introduces its all- electric IRb 6 and IRb 60 robots, designed for automatic grinding operations.

1974
Hitachi uses touch and force sensing with its Hi-T-Hand robot, allowing the robot hand to guide pins into holes.

1974
The Robotics Industries Association is founded.

1975
Cincinnati Milacron introduces its first T3 robot for drilling applications.

The ASEA 60kg robot is the first robot installed in an iron foundry; the Cincinnati Milacron T3 becomes the first robot to be used in the aerospace industry.

1976
The Trallfa spray-painting robot is adapted for arc welding at the British agricultural implement firm of Ransome, Sims and Jefferies.

1976
Remote Center Compliance evolves from research at Charles Stark Draper Labs, Cambridge, Mass. Dynamics of part mating are developed, allowing robots to line up parts with holes both laterally and rotationally.

1976
The robotic spacecraft "Viking" (United States) lands on the Martian surface.

1977
California Institute of Technology's Jet Propulsion Laboratory (JPL) demonstrates a robotic hand-eye system can be integrated with a self-propelled vehicle for planetary exploration. (Mars Rover)

1977
The British Robotics Association (BRA) is founded.

1978
The first PUMA (Programmable Universal Assembly) robot is developed by Unimation for General Motors.

1978
The Machine Intelligence Company is organized by Charles A. Rosen and associates.

1979
Japan introduces the SCARA (Selective Compliance Assembly Robot Arm); Digital Electronic Automation (DEA) of Turin, Italy, introduces the PRAGMA robot, which is licensed to General Motors.

1980
Robotics languages are developed to ease programming bottlenecks.

1981
IBM enters the robotics field with its 7535 and 7565 Manufacturing Systems.

1982
The Pedesco robot (Pedesco, Scarborough, Ontario) is used to clean up after a nuclear fuel spill at an atomic power plant. A task too dangerous for direct human contact.

1982
Stan Mintz and five co-employees of Hewlett-Packard Company left to form Intelledex Corporation, a manufacturer of light assembly robots, for such tasks as installing integrated circuits.

1981-1984
Rehabilitation robots are enhanced by mobility, voice communication, and safety factors. Greater emphasis is placed on machine vision, tactile sensors, and languages. Battlefield and security robots are developed.

1983
Westinghouse Electric Corporation buys Unimation, Inc., which becomes part of its factory automation enterprise. Westinghouse later sells Unimation to AEG of Pennsylvania.

1984
Robot Defense Systems introduces the Prowler ("Programmable Robot Observer with Local Enemy Response"), the first in a series of battlefield robots.

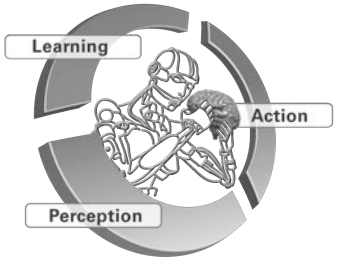
1984
Intelledex Corporation introduces the Model 695 lite assembly robot, based on the Intel 8086 and 8087 microprocessor chips. Its software is called Robot Basic, a specialized version of Microsoft's Basic.

1993
The University of Michigan's CARMEL robot wins first place at the 1992 Robot Competition sponsored by the American Association for Artificial Intelligence (AAAI). CARMEL stands for computer-aided robotics for maintenance, emergency, and life support. The SRI International's robot "FLAKEY" wins second place. Both microcomputer- controlled machines use ultrasonic sonar sensors.

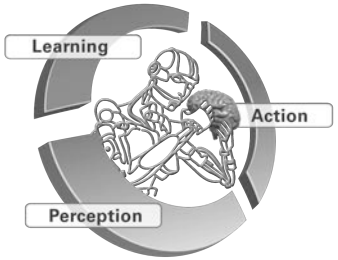
1997
The Honda Humanoid Robot is introduced

2002
The Roomba Robot is the first household robot to be sold more than one million times

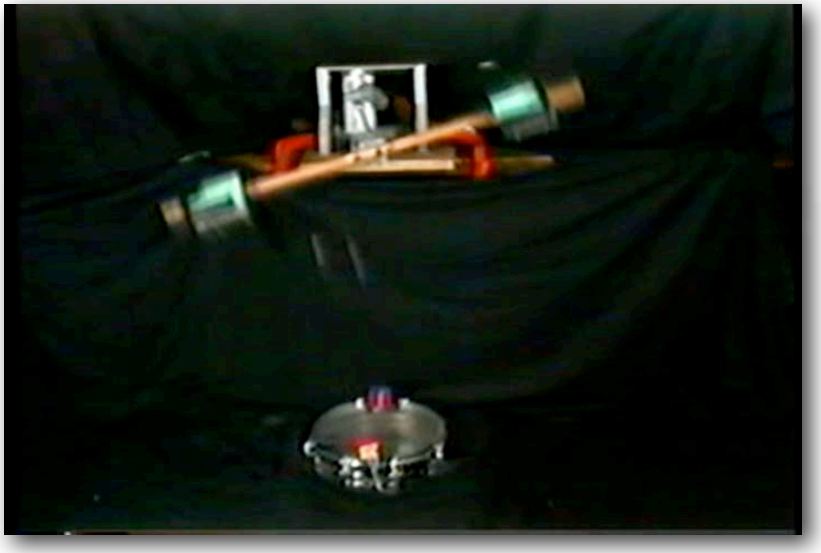
2005
Autonomous Car Navigation in complex terrain in the DARPA Grand Challenge

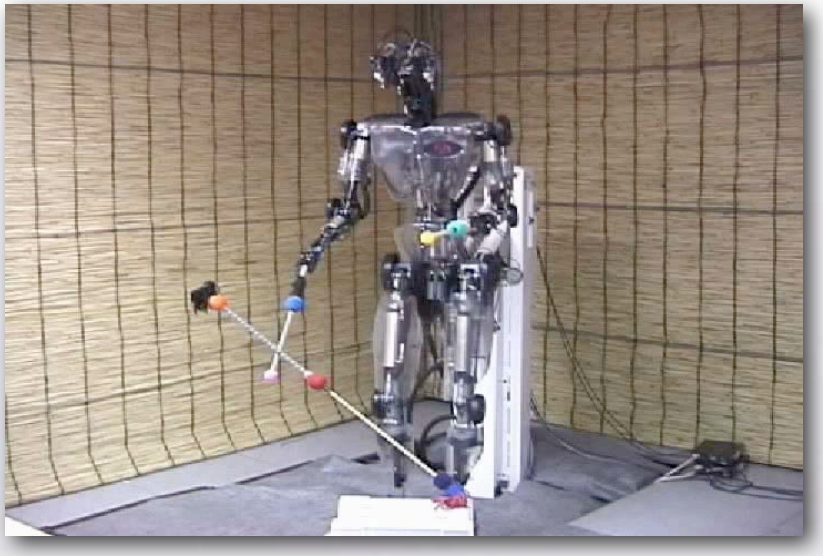


Robotics–What We Might Want



Some History of Robot Juggling

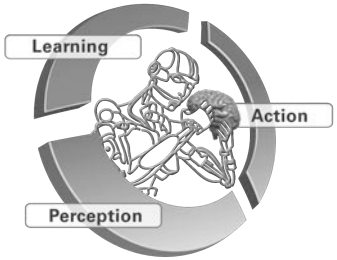




The Flying Machine Arena
Quadrocopter Ball Juggling

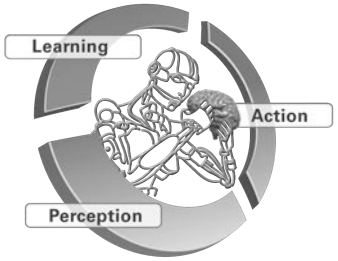
IDSC

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

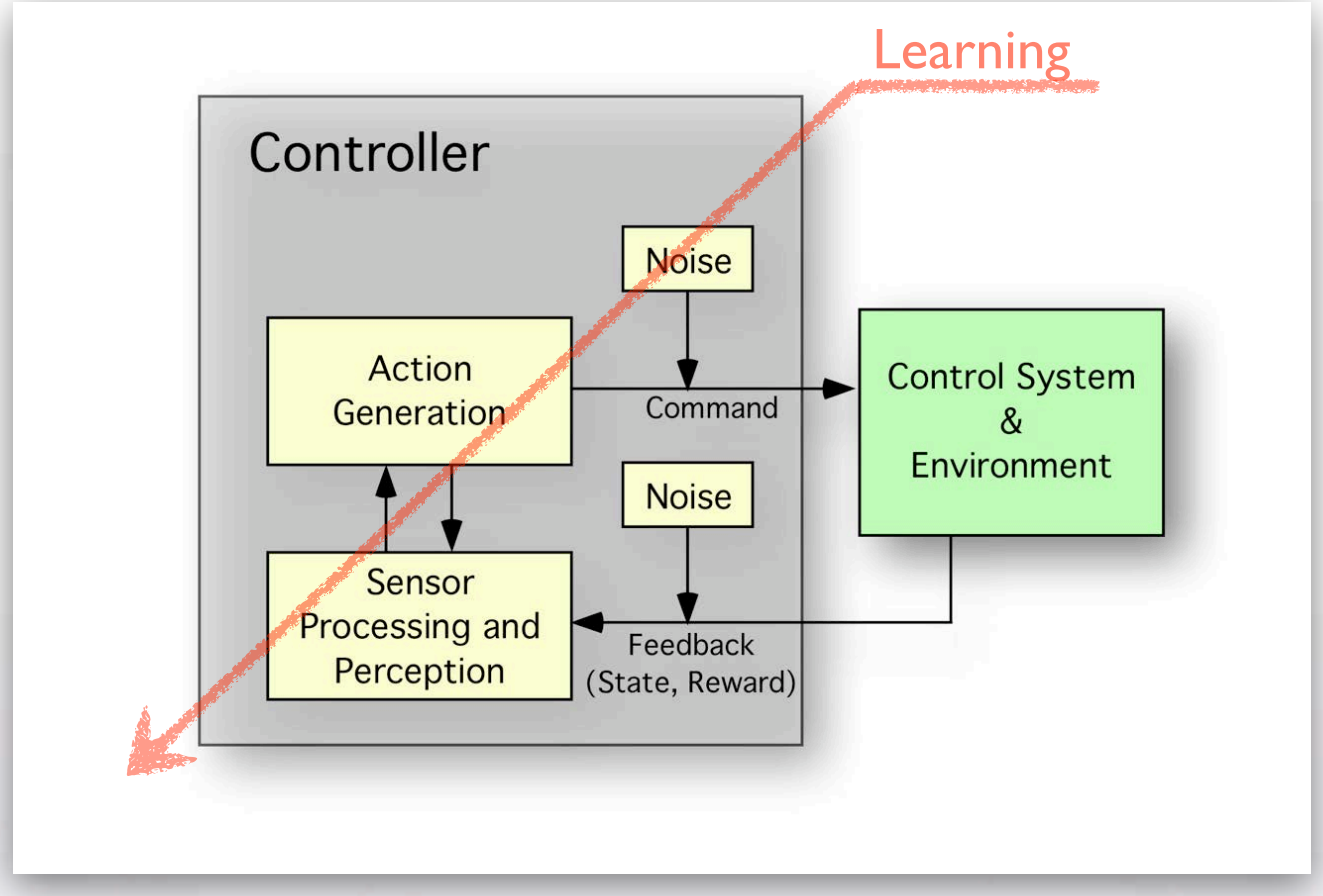


Outline

- A Bit of Robotics History
- Foundations of Control
- Adaptive Control
- Learning Control
  - Model-based Robot Learning

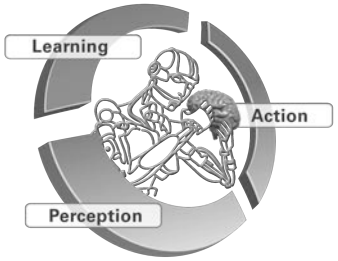


Control Diagrams: Perception-Action-Learning



System Model: $\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}, \mathbf{u}, t, \varepsilon_{\mathbf{x}}\right)$

Observation Model: $\mathbf{y} = \mathbf{h}\left(\mathbf{x}, \mathbf{u}, t, \varepsilon_{\mathbf{y}}\right)$

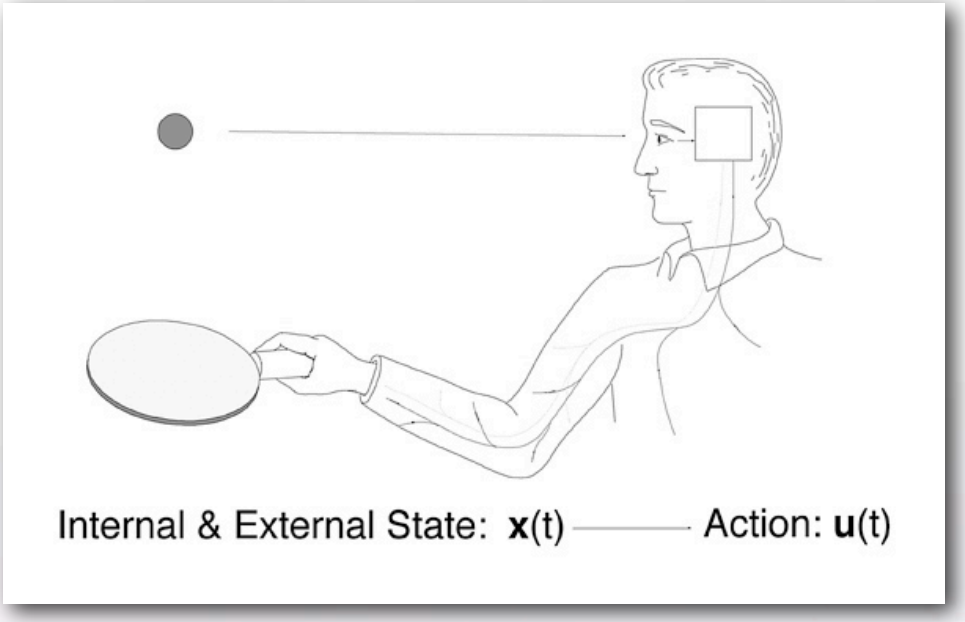


Control Policies

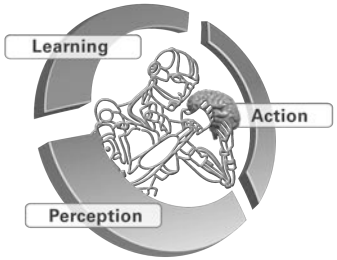
- The General Goal of Control:

Control Policies

$$\mathbf{u}(t) = \pi(\mathbf{x}(t), t, \alpha)$$



But how should control policies be represented?

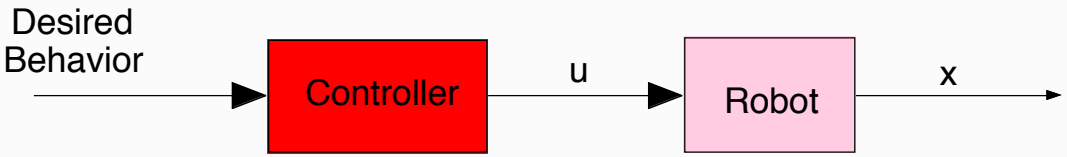


Representing Control Policies

- Feedforward Control:
  - Control policy **does not** receive feedback from the robot/environment

Open Loop Control

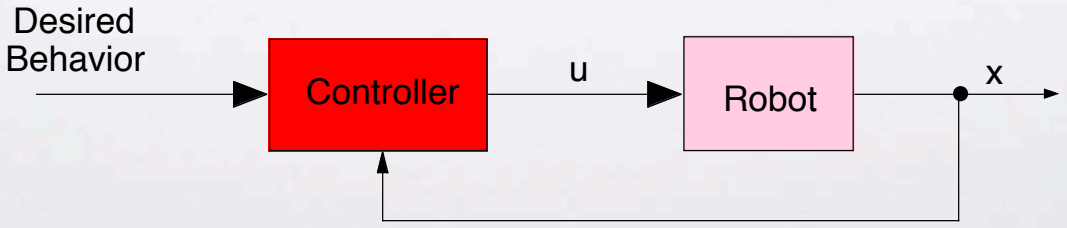
$$\mathbf{u} = \pi(\mathbf{x}, \alpha, t) = \pi(\alpha, t)$$

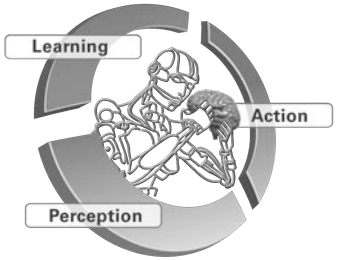


- Feedback Control
  - Control policy **does** receive feedback from the robot/environment

Closed Loop Control

$$\mathbf{u} = \pi(\mathbf{x}, \alpha, t)$$



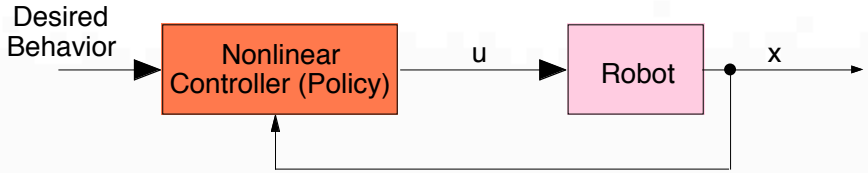


Representing Control Policies: Types of Feedback Control

- General Feedback Control:

Feedback Control

$$\mathbf{u} = \pi(\mathbf{x}, \alpha, t)$$



- Negative Feedback Control

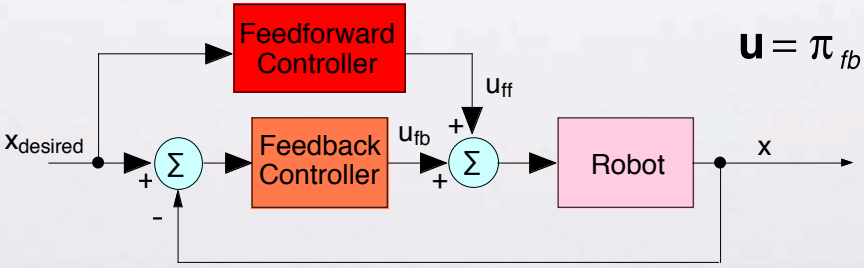
Negative Feedback Control

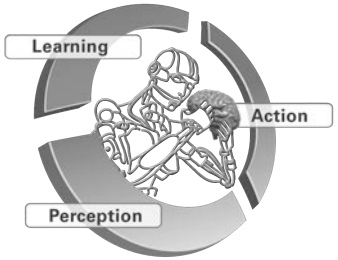
$$\mathbf{u} = \pi(\mathbf{x} - \mathbf{x}_{des}, \alpha, t)$$

- Negative Feedback and Feedforward Control

Neg. Feedback & Feedforward Control

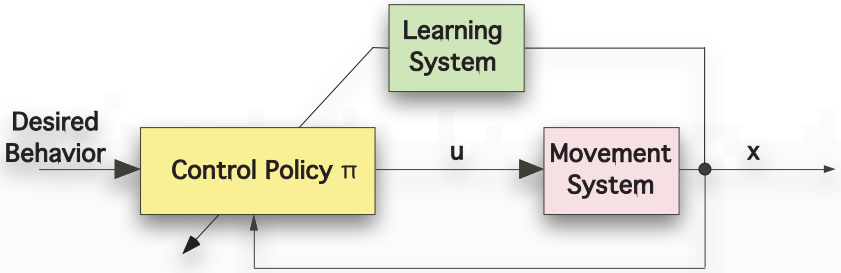
$$\mathbf{u} = \pi_{fb}(\mathbf{x} - \mathbf{x}_{des}, \alpha, t) + \pi_{ff}(\mathbf{x}_{des}, \alpha, t)$$

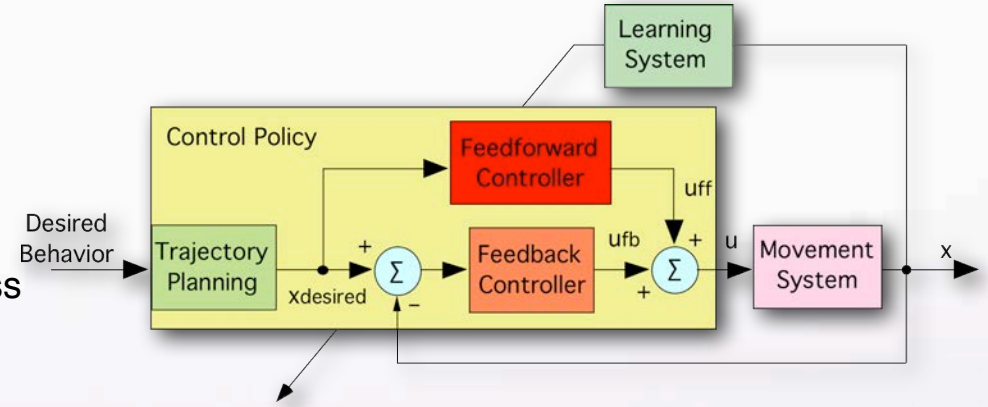


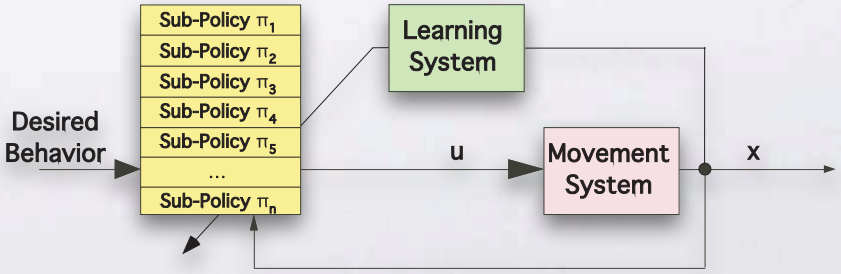


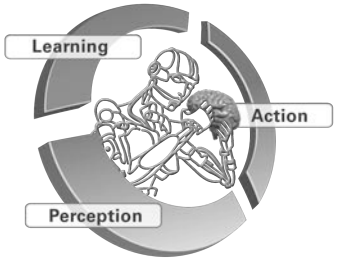
Representing Control Policies: Variations of Closed-Loop Control

- Direct Control:
  - policy creates directly motor commands
    - Pros: Very general representation
    - Cons: Hard to find these control policies, hard to re-use (generalization)
- Indirect Control
  - policy creates kinematic trajectory plans, and converts them to motor commands
    - Pros: easier to re-use
    - Cons: more pre-structure required, less flexible in representational power
- Modularization
  - Motor primitives are used to generate complex behaviors from smaller pieces
    - Pros: easier generalization and re-use
    - Cons: less representational power, how to determine/learn the modularization?









Linear Negative Feedback Control

- Proportional-Derivative-Integral (PID) Control:

$$\mathbf{u}_{fb} = \mathbf{u}_P + \mathbf{u}_D + \mathbf{u}_I$$

  - Proportional Control ("Position Error")

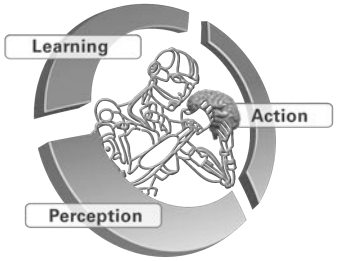
$$\mathbf{u}_P = \pi(\mathbf{x} - \mathbf{x}_{des}, \alpha, t) = \mathbf{K}_P(\mathbf{x}_{des}(t) - \mathbf{x}(t))$$

  - Derivative Control ("Damping")

$$\mathbf{u}_D = \pi(\dot{\mathbf{x}} - \dot{\mathbf{x}}_{des}, \alpha, t) = \mathbf{K}_D(\dot{\mathbf{x}}_{des}(t) - \dot{\mathbf{x}}(t))$$

  - Integral Control ("Steady State Error")

$$\mathbf{u}_I(t) = \mathbf{K}_I \int_{\tau=0}^{\tau=t} (\mathbf{x}_{des}(t) - \mathbf{x}(t))\, dt$$



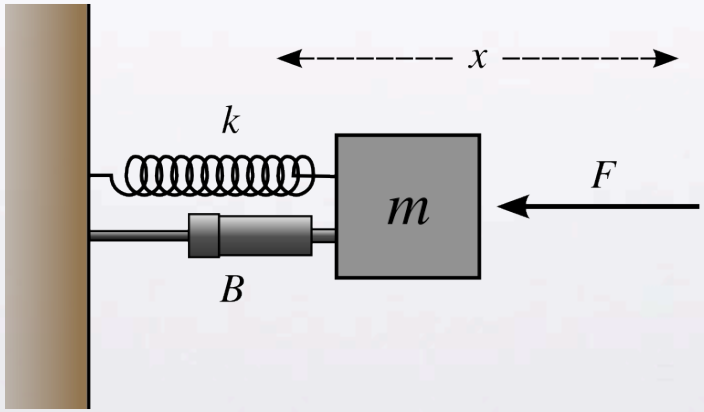
Model-based Feedforward Control

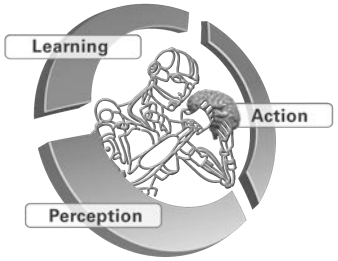
- Rigid-Body Dynamics: A General Modeling Framework for most robots

$$\mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \tau$$

  - One of the simplest examples: Linear mass-spring-damper

$$m\ddot{x} + B\dot{x} + k(x - x_0) = F$$





Model-based Feedforward Control

- Some notation:
  - Inverse dynamics:

$$\mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \tau$$

  - Forward dynamics:

$$\ddot{\mathbf{q}} = \mathbf{B}^{-1}(\mathbf{q})\left(\tau - \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} - \mathbf{G}(\mathbf{q})\right)$$

  - Control Affine Dynamics:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x})\mathbf{u}$$

  - Control Law:

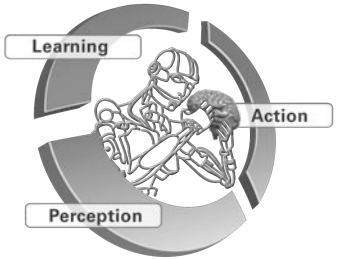
$$\mathbf{u} = \mathbf{u}_{fb} + \mathbf{u}_{ff}$$

with, for example, Computed Torque Control:

$$\mathbf{u}_{ff} = \mathbf{B}(\mathbf{q}_{des})\ddot{\mathbf{q}}_{des} + \mathbf{C}(\mathbf{q}_{des},\dot{\mathbf{q}}_{des})\dot{\mathbf{q}}_{des} + \mathbf{G}(\mathbf{q}_{des})$$

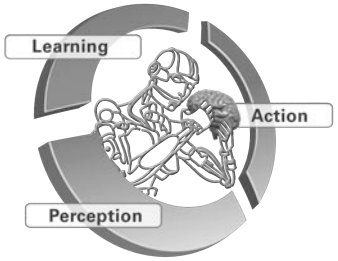
$$\mathbf{u}_{fb} = \mathbf{K}_P(\mathbf{q}_{des} - \mathbf{q}) + \mathbf{K}_D(\dot{\mathbf{q}}_{des} - \dot{\mathbf{q}})$$

$\mathbf{K}_P, \mathbf{K}_D$ are positive definite

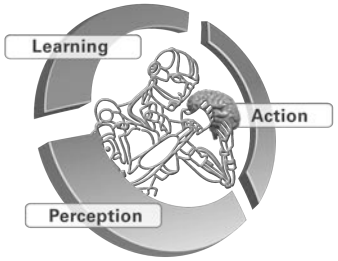


Model-based Control

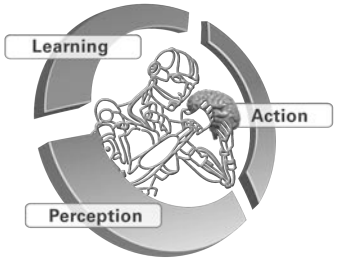
- Some Important Properties
  - most industrial robots only use feedback control
  - rigid body dynamics models require special software to derive, as equations easily go over 10-100 pages
  - compliant control requires model-based control and accurate models
  - negative feedback control is always needed for error/perturbation rejection
  - damping is very important to ensure stability
  - modern, compliant robots require model-based control



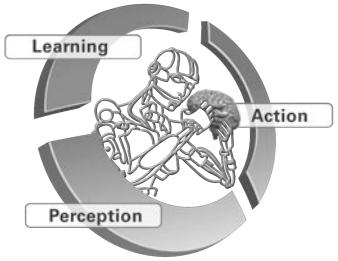
Example: Model-based Control of a Robot Dog



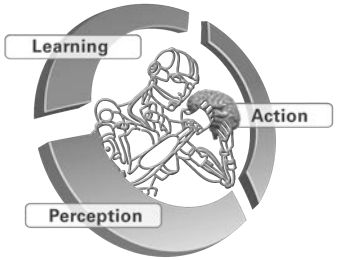
Example: Model-based Control of a Robot Dog



Example: Model-based Control of a Robot Dog

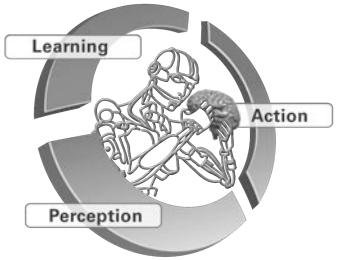


Example: Model-based Control of a Robot Dog



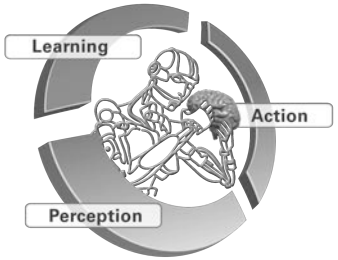
Outline

- A Bit of Robotics History
- Foundations of Control
- Adaptive Control
- Learning Control
  - Model-based Robot Learning



Adaptive Control

- Terminology
  - Adaptive Control, in control theory, denotes an adaptation (=learning) process where the control system IS NOT PERMITTED TO FAIL
  - This is in contrast to Learning Control, which does allow for trial-and-error learning with failure
  - Emphasis on the closed-loop system's stability, which really matters the most in robotics
- Basic Steps of Adaptive Control
  - Characterize the desired behavior of the closed loop system
  - Determine a suitable control law with adjustable parameters
  - Find a mechanism for adjusting the parameters
  - Implement the control law



Model-Reference Adaptive Control

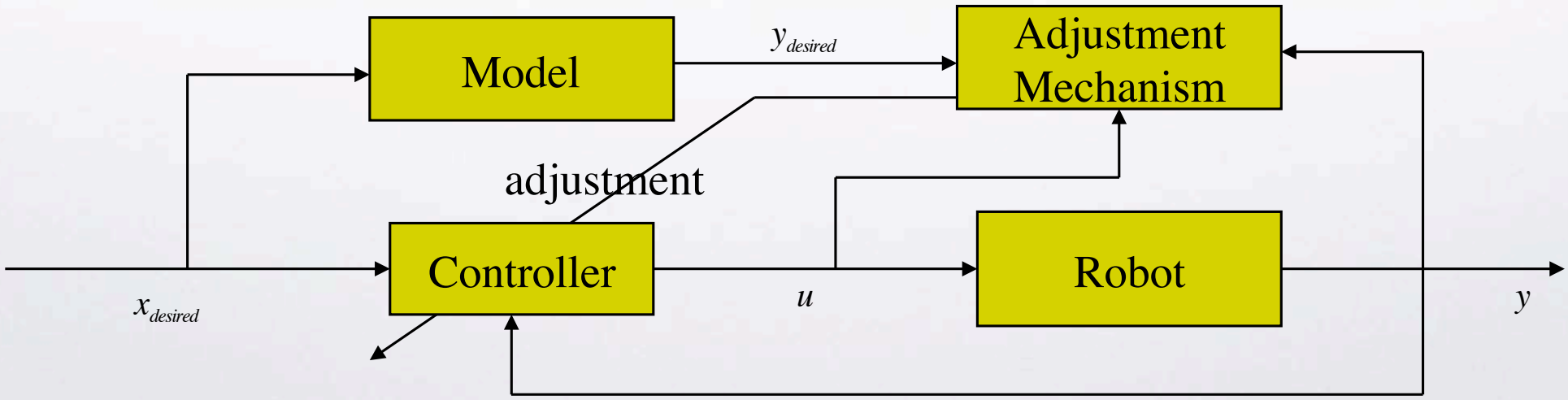
- Performance is supposed to correspond to a particular reference model
  - for example:

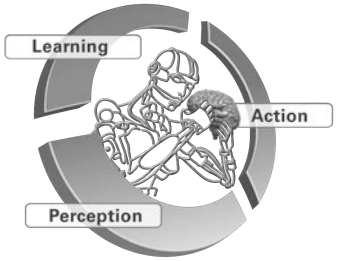
$$y_{desired} = x_{desired}$$

$$or$$

$$\dot{y}_{desired} = \alpha\left(x_{desired} - y_{desired}\right)$$

  - this defines a cost function, which allows to adjust parameters, e.g., by gradient descent





Example: Deriving A Simple Adaptive Control Law

- Consider a generic control affine system

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}$$

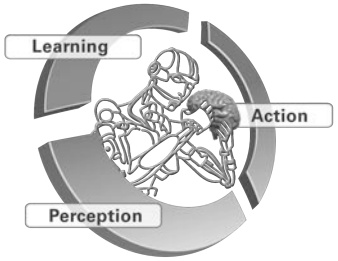
- Let's make it even simpler

$$\dot{x} = f(x) + u$$

- Control objective: accurately follow x_d

- Assume that f is unknown and needs to be estimated by a learning process. Thus, we can formulated a model-based control law:

$$u = -\hat{f}(x) + \dot{x}_d - k(x - x_d)$$



Example: Deriving A Simple Adaptive Control Law

- Assume that f can be modeled accurately by a linear system

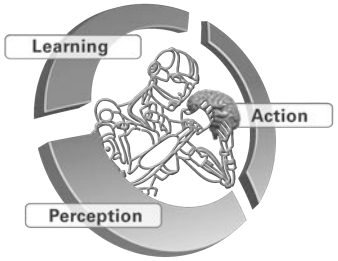
$$f(x) = \theta x$$

$$\hat{f}(x) = \hat{\theta} x$$

- such that our control law becomes:

$$u = -\hat{\theta} x + \dot{x}_d - k(x - x_d)$$

- The goal of model-reference adaptive control is to adjust the open parameter and the control law such that the system is ALWAYS stable



Example: Deriving A Simple Adaptive Control Law

- After inserting the control law, the system dynamics is:

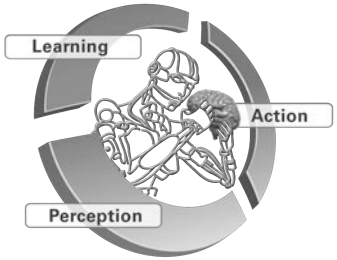
$$\dot{x} = f(x) + u$$
$$= \theta x - \hat{\theta} x + \dot{x}_d - k(x - x_d)$$

- Define errors:

$$e = x_d - x$$
$$\tilde{\theta} = \theta - \hat{\theta}$$

- Insert into system dynamics to obtain erro dynamics:

$$\dot{x} = \theta x - \hat{\theta} x + \dot{x}_d - k(x - x_d)$$
$$= \tilde{\theta} x + \dot{x}_d + ke$$
$$\dot{e} = -ke - \tilde{\theta} x$$



Example: Deriving A Simple Adaptive Control Law

- Define a Lyapunov function:
  - Control stability is guaranteed if dV/dt < 0

$$V = \frac{1}{2}e^2 + \frac{1}{2}\tilde{\theta}\ \Gamma^{-1}\tilde{\theta}$$

- Take time derivative of Lyapunov function:

$$\begin{aligned}\dot{V} &= e\dot{e} + \tilde{\theta}\ \Gamma^{-1}\dot{\tilde{\theta}} \\ &= e\dot{e} - \tilde{\theta}\ \Gamma^{-1}\dot{\hat{\theta}} \\ &= e\left(-x\ \tilde{\theta} - ke\right) - \tilde{\theta}\ \Gamma^{-1}\dot{\hat{\theta}} \\ &= -ex\tilde{\theta} - ke^2 - \tilde{\theta}\ \Gamma^{-1}\dot{\hat{\theta}}\end{aligned}$$

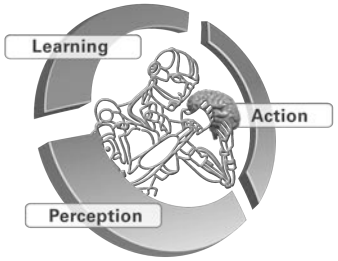
- Choose open parameters such that dV/dt<0:

$$-ex\tilde{\theta} - \tilde{\theta}\ \Gamma^{-1}\dot{\hat{\theta}} = 0$$

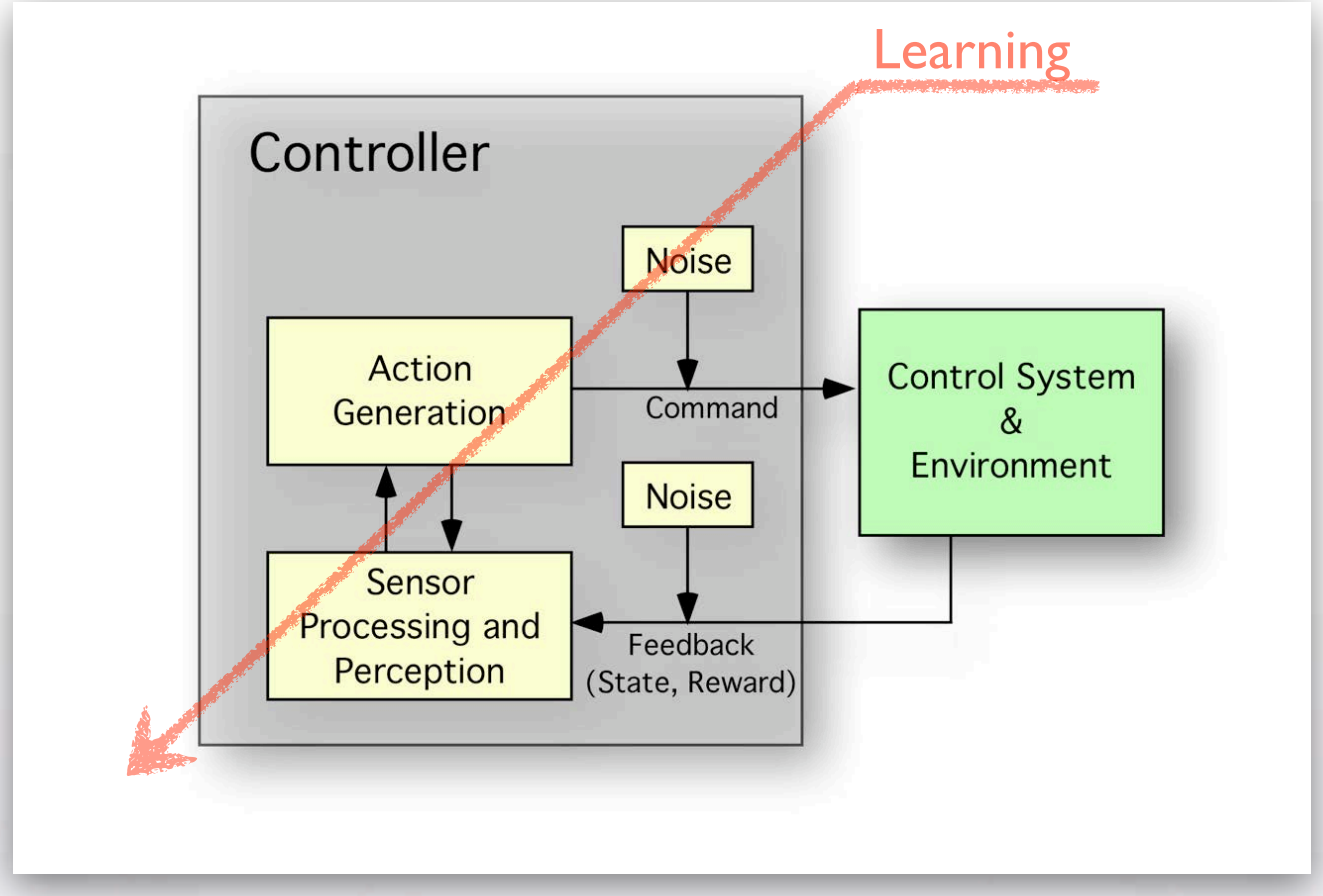
$$\Rightarrow$$

$$\dot{\hat{\theta}} = -\Gamma ex$$

This parameter adaptation law guarantees stability!

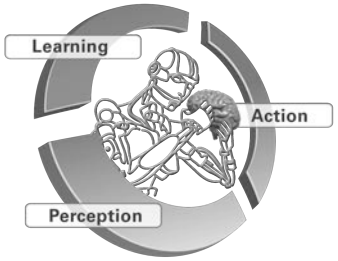


Adaptive Control Cares About The Closed Loop System!

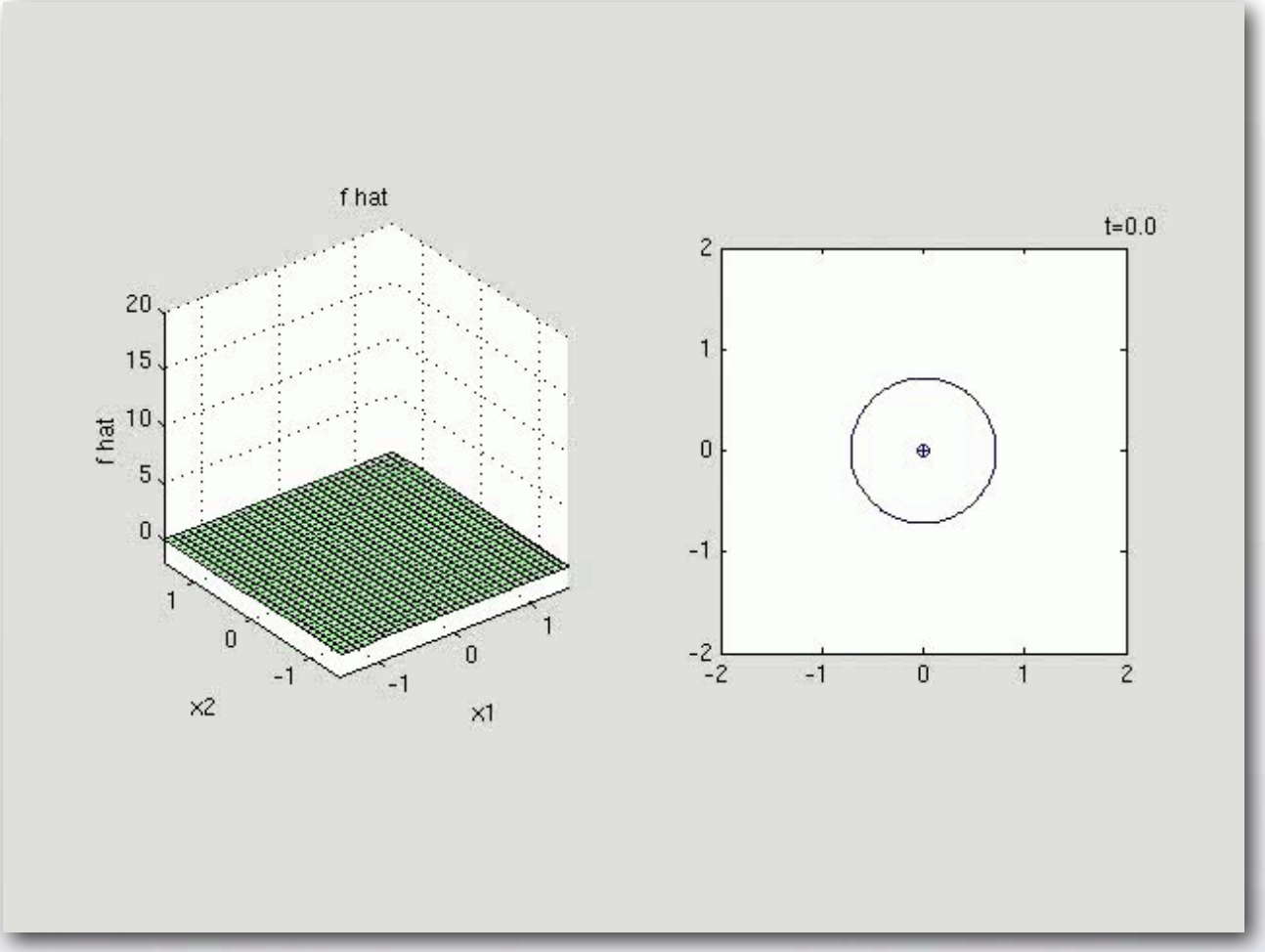


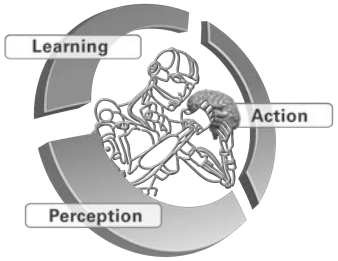
System Model: $\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}, \mathbf{u}, t, \varepsilon_{\mathbf{x}}\right)$

Observation Model: $\mathbf{y} = \mathbf{h}\left(\mathbf{x}, \mathbf{u}, t, \varepsilon_{\mathbf{y}}\right)$



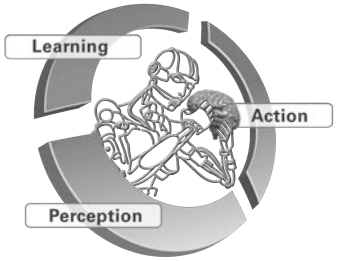
A Nonlinear Example





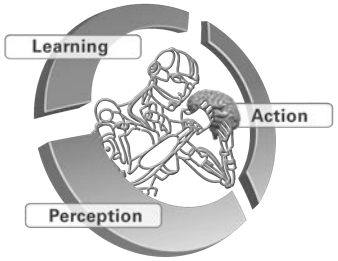
Example of Nonlinear Adaptive Control

Whole Arm Manipulator Catching

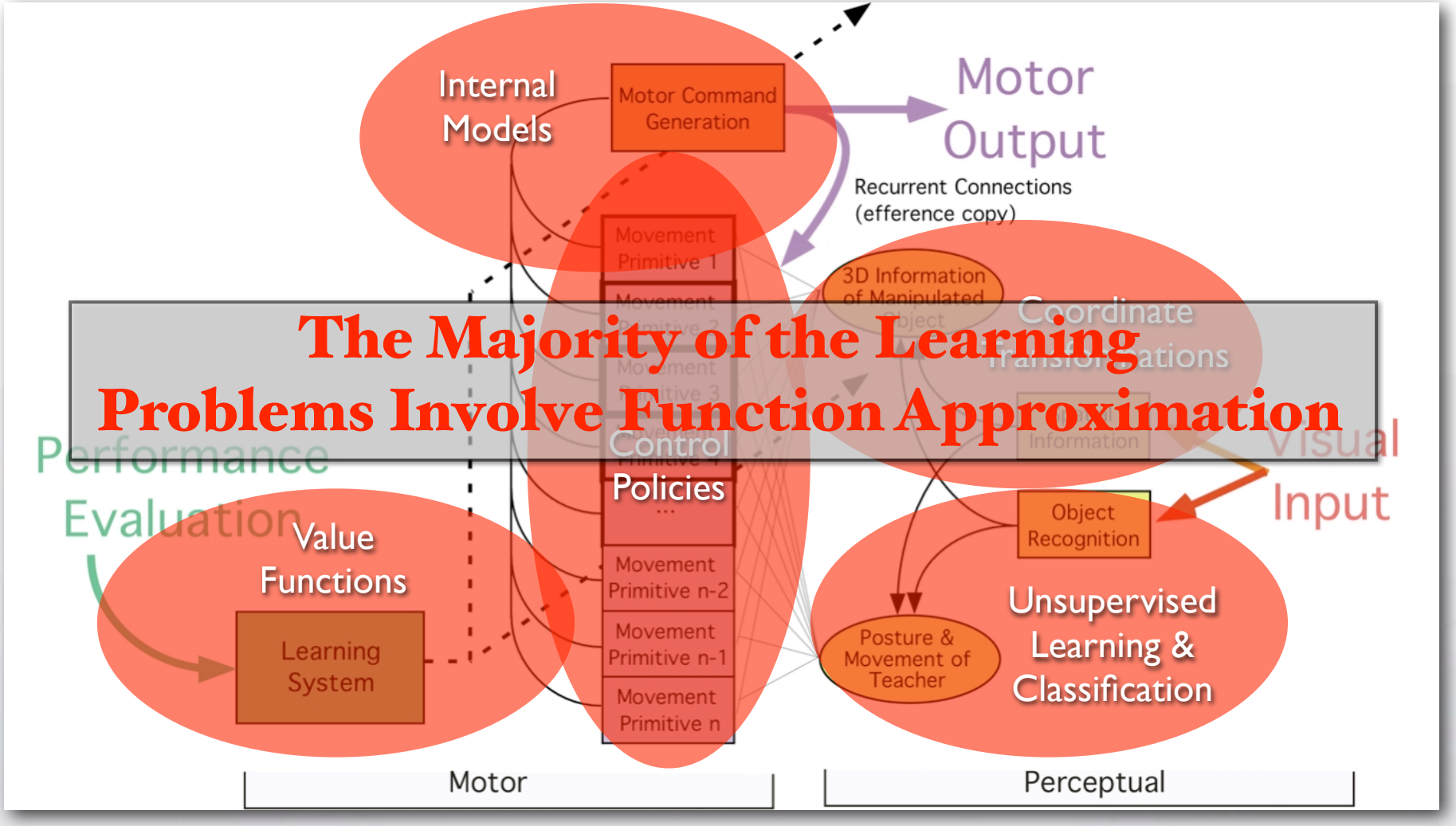


Outline

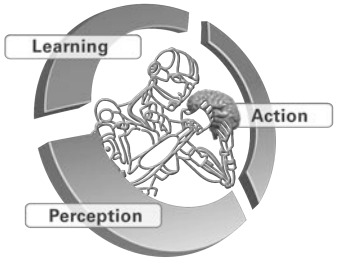
- A Bit of Robotics History
- Foundations of Control
- Adaptive Control
- Learning Control
  - Model-based Robot Learning



What Needs to Be Learned in Learning Control?



The Majority of the Learning Problems Involve Function Approximation



Learning Internal Models

- Forward Models
  - models the causal functional relationship
  - for example:

$$\mathbf{y} = \mathbf{f}(\mathbf{x})$$

$$\ddot{\mathbf{q}} = \mathbf{B}^{-1}(\mathbf{q})\left(\tau - \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} - \mathbf{G}(\mathbf{q})\right)$$

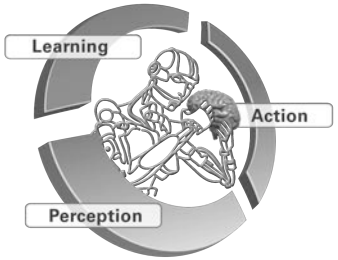
- Inverse Models
  - models the inverse of the causal functional relationship

$$\mathbf{x} = \mathbf{f}^{-1}(\mathbf{y})$$

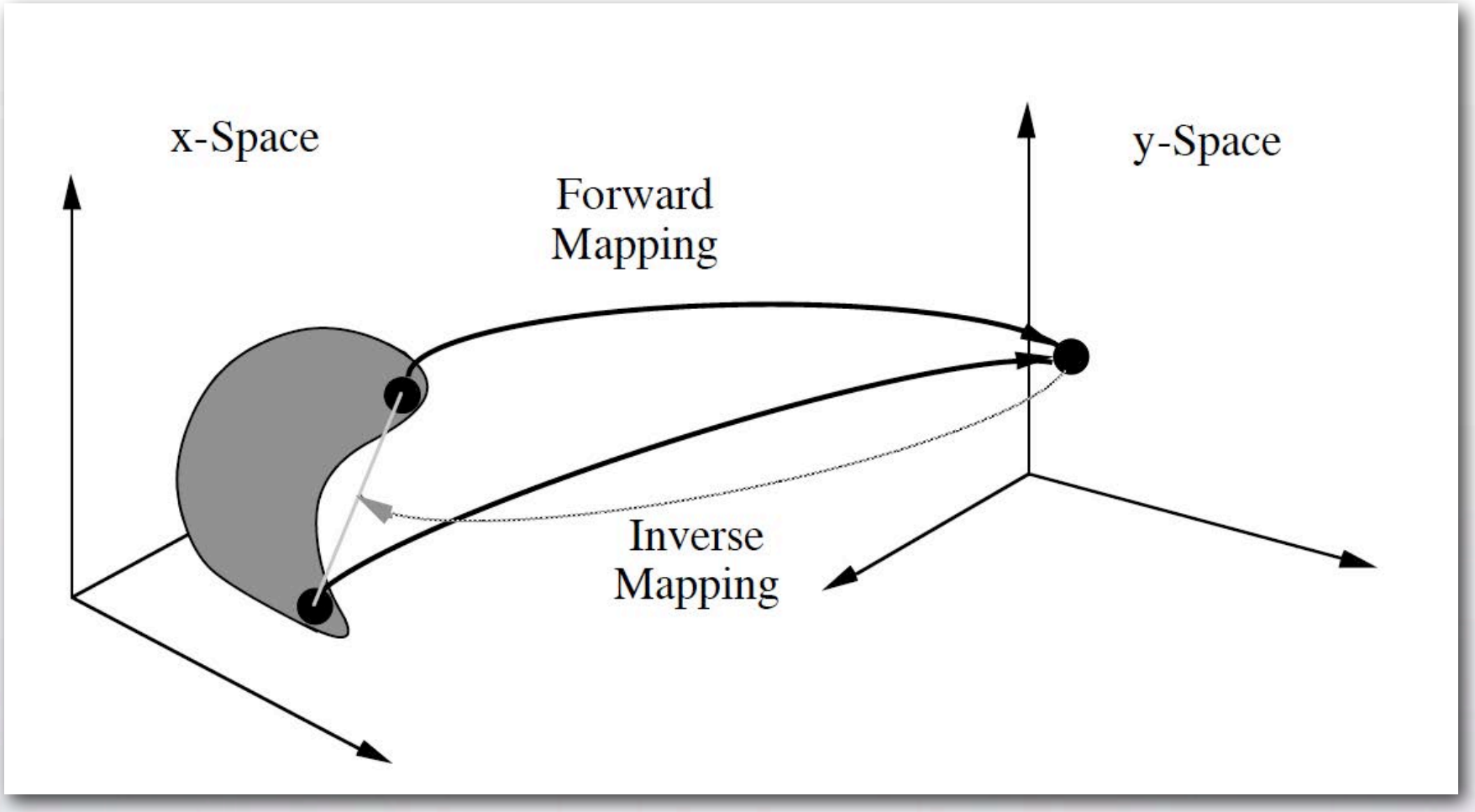
  - for example:

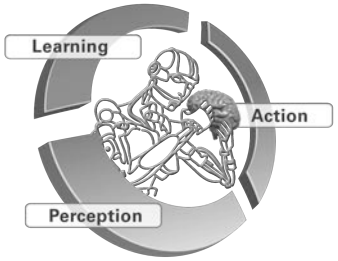
$$\mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \tau$$

  - NOTE: inverse models are not necessarily functions any more!



Inverse Models May Not Be Trivially Learnable



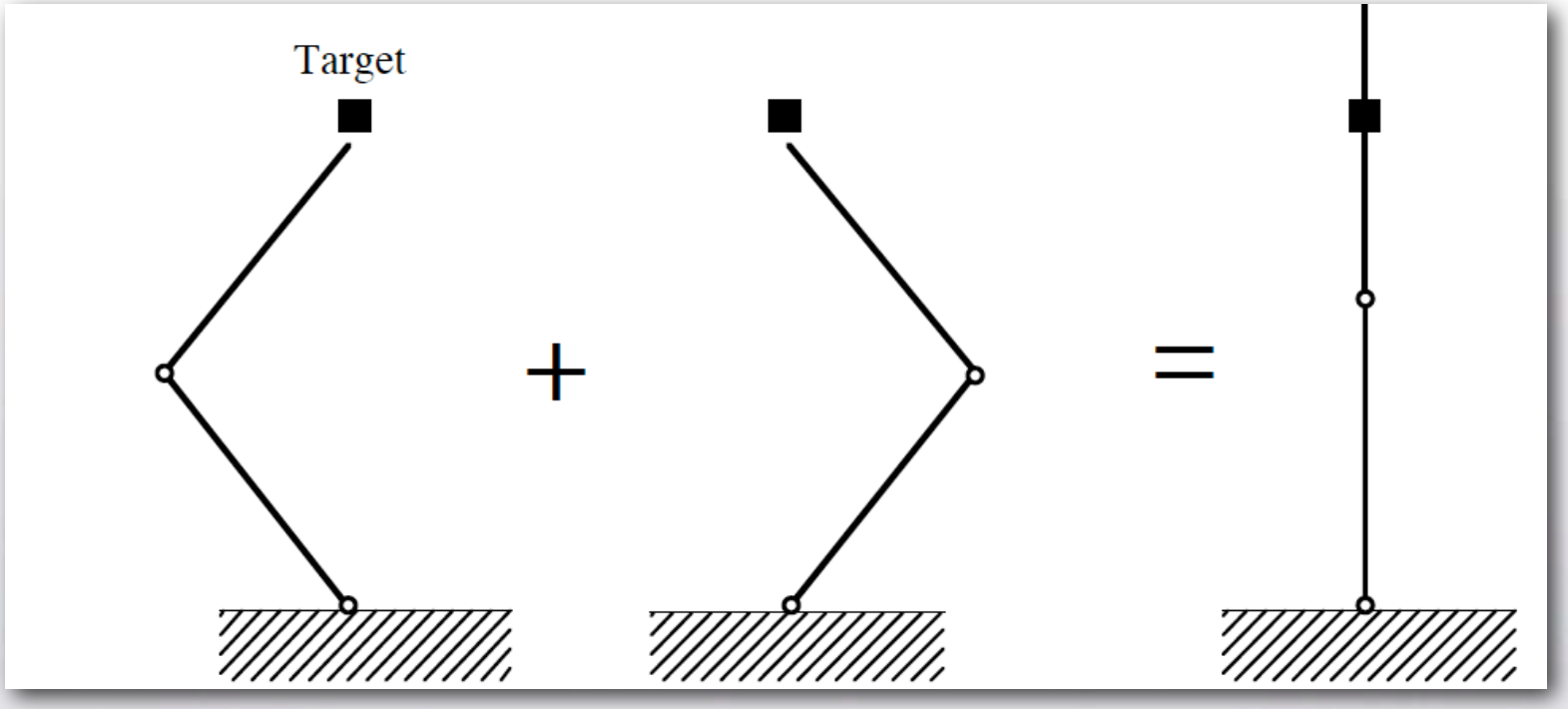


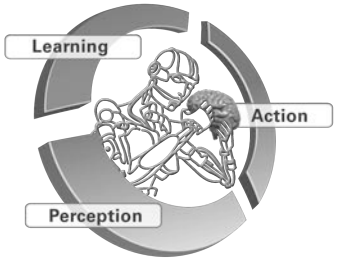
Inverse Models May Not Be Trivially Learnable

$$\mathbf{t} = \mathbf{f}\left(\theta_1^1, \theta_2^1\right)$$

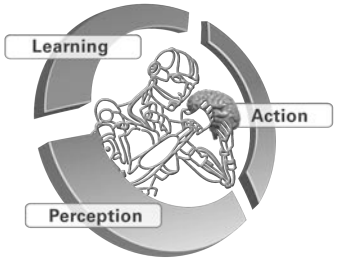
$$\mathbf{t} = \mathbf{f}\left(\theta_1^2, \theta_2^2\right)$$

what is $\mathbf{f}^{-1}(\mathbf{t})$?





... continued in the next lecture.



A Cool Robot Movie