IVR: Sensing the World

Keypoints:

- Why robots need sensing
- Factors that affect sensing capability
- Contact sensing
- Proximity and range sensing
- Sensing light

Why robots need sensing

For a robot to act successfully in the real world it needs to be able to perceive the world, and itself in the world.

Can consider sensing tasks in two broad classes:

Finding out **what** is out there: e.g. is there a goal; is this a team-mate; is there danger? = *Recognition*

Finding out **where** things are: e.g. where is the ball and how can I get to it; where is the cliff-edge and how can I avoid it? = *Location*

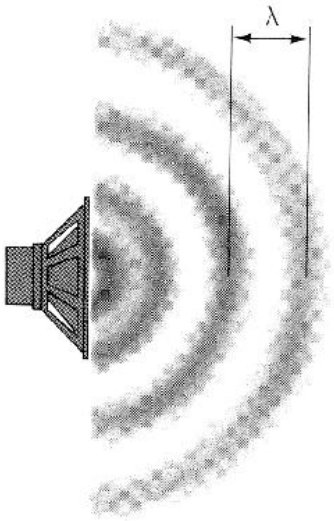
But note that this need not be explicit knowledge

Sensing capability

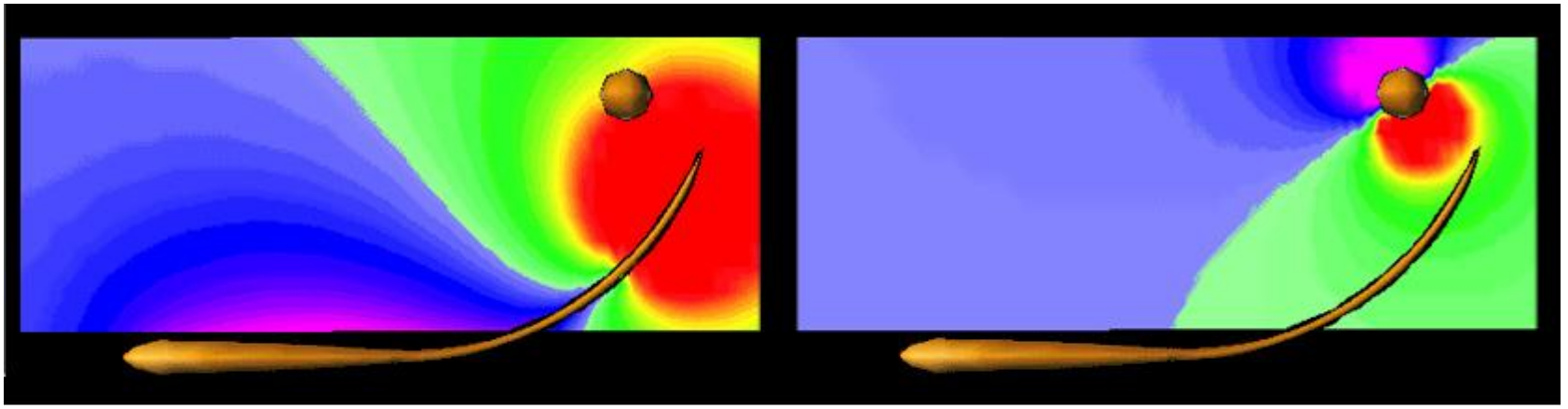
depends on a number of factors:

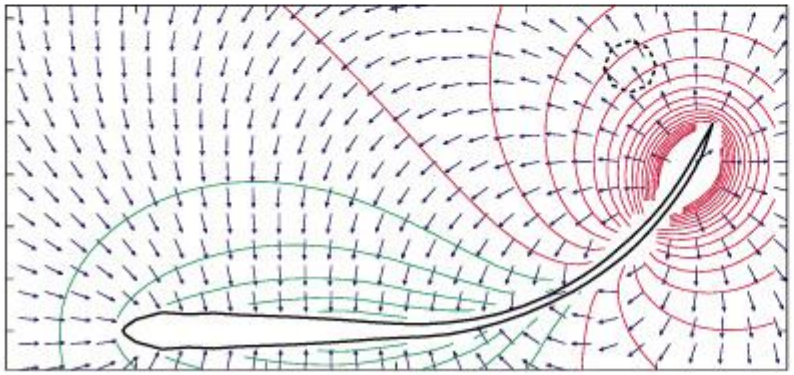
1. What signals are available?

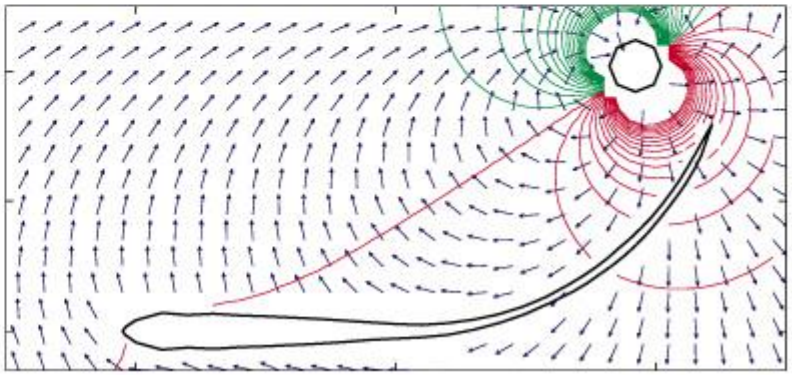
Light Pressure & Sound Chemicals



N.B. Many more signals in world than humans usually sense: e.g. Electric fish generate electric field and detect distortion







Sensing capability

1 What signals are available?

2 What are the capabilities of the sensors?

Distance	Contact	Internal
Vision	Taste	Balance
Hearing	Pressure	Actuator position and movement
Smell	Temperature	Pain or damage

Note this differs across animals: e.g. Bees see ultraviolet light

Need to choose what to build in to robot – options and costs

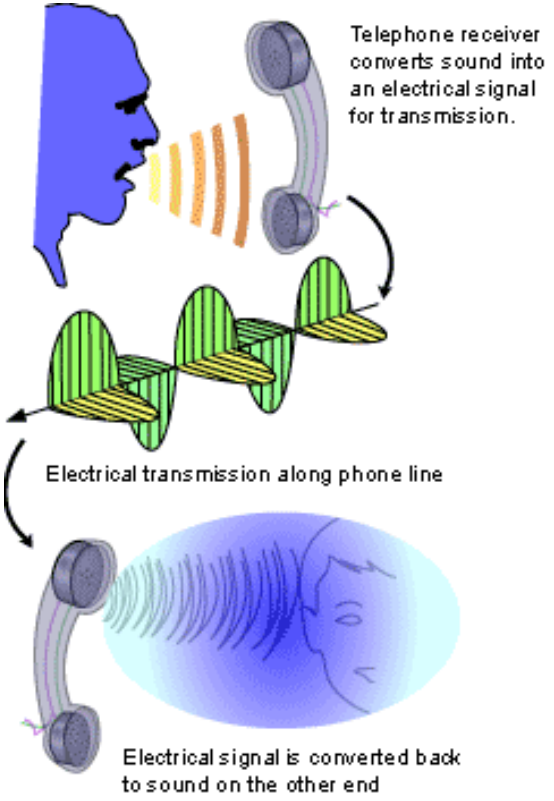
Visible

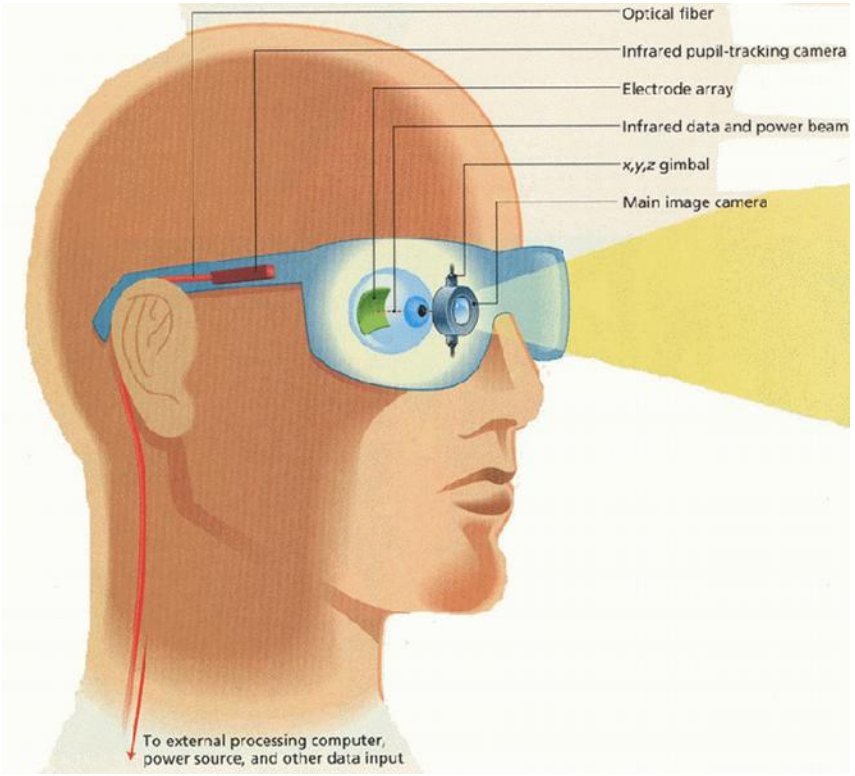
Ultraviolet

More like a target?

Sensors perform transduction

Transduction: transformation of energy from one form to another (typically, into electrical signals)





Sensors perform transduction

Sensor characteristics mean there is rarely an isomorphic mapping between the environment and the internal signal, e.g:

- Most transducers have a limited range
- Most transducers have a limited resolution, accuracy, and repeatability
- Most transducers have lags or sampling delays
- Many transducers have a non-linear response
- Biological transducers are often adaptive
- Good sensors are usually expensive in cost, power, size…

Sensing capability

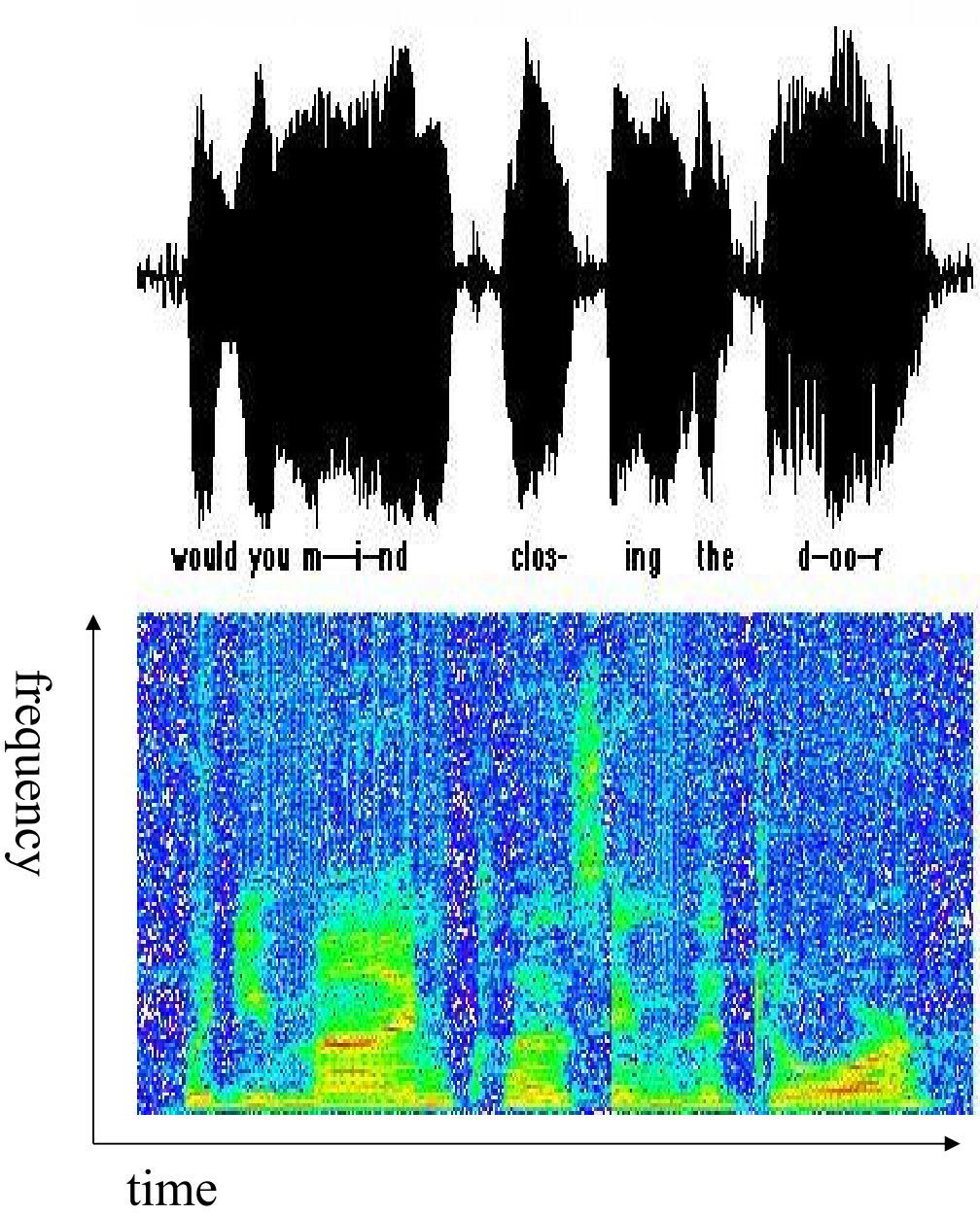
1 What signals are available?

2 What are the capabilities of the sensors?

3 What processing is taking place?

E.g. extracting useful information from a sound signal is difficult:

- Sound sources cause air vibration
- Diaphragm (ear drum or microphone) has complex pattern of vibration in response to sound
- Usually analysed by separating frequencies and grouping through harmonic/ temporal cues



Artificial sensing in neuroprosthetics

- Cochlear implant
- Retinal implant
- Cortical/brainstem implants
- Foot-drop neuro-prosthesis
- Touch in handprosthesis

Sensing capability

depends on a number of factors:

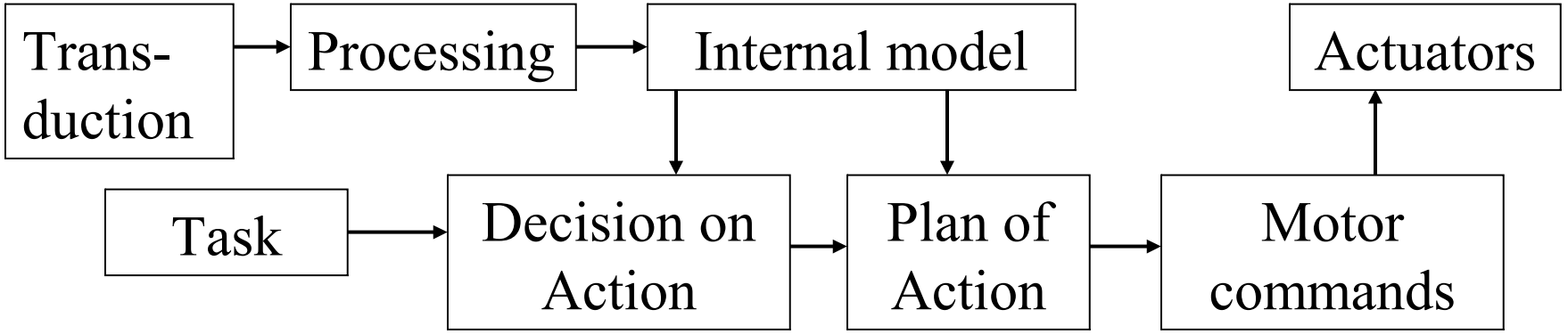
1 What signals are available?

2 What are the capabilities of our sensors?

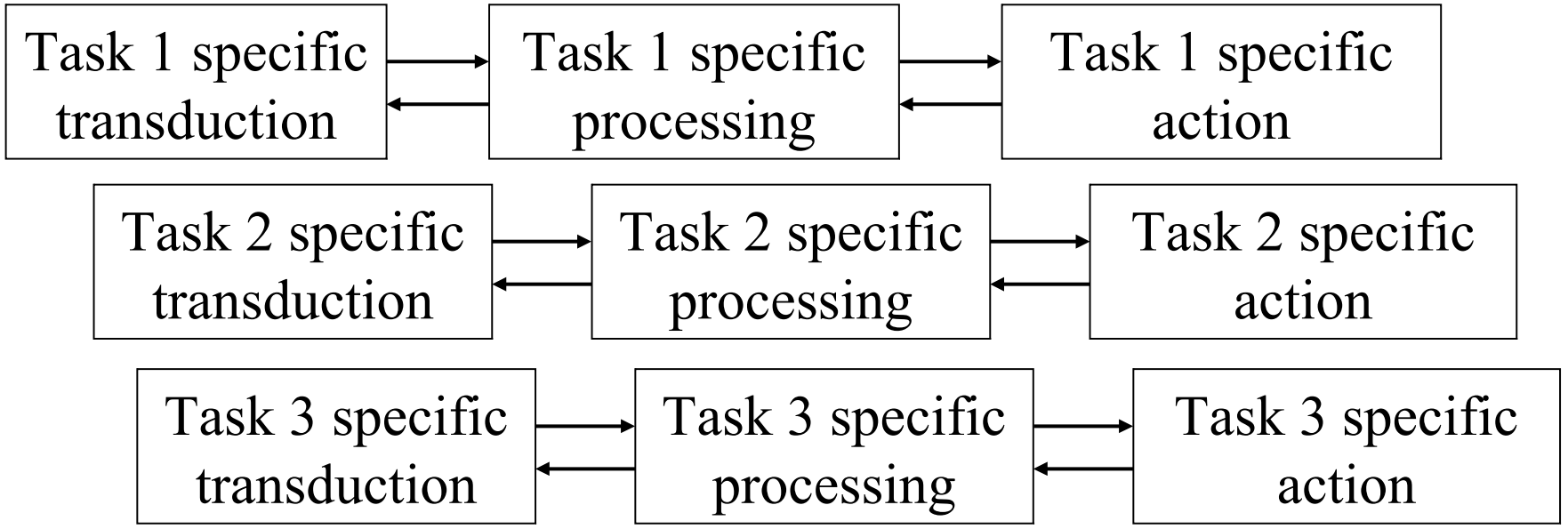
3 What processing is taking place?

4 What is the task?

‘Classical’ view



Alternative view



Affordances

"Perceivable potentialities of the environment for an action"

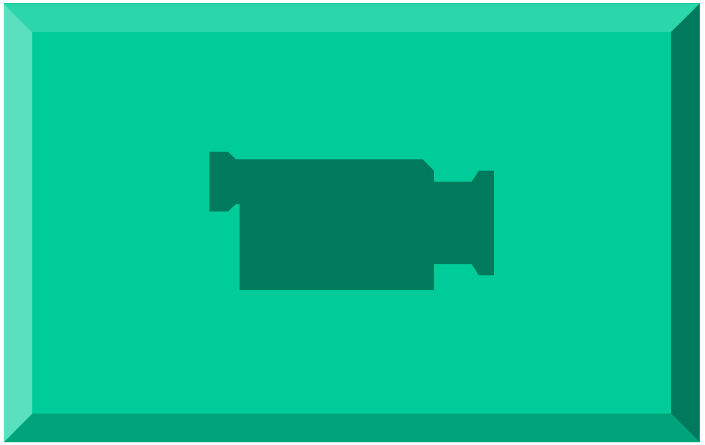
Scan scene, build surface model, analyse surfaces, find flat one near feet

vs.

Use "flat surface near feet" special detector

First is traditional, second affordance-based: sensors tuned for exactly what is needed for the task

Keep close count of how many times the white team pass their ball



(Please remain silent till the end of the video clip)

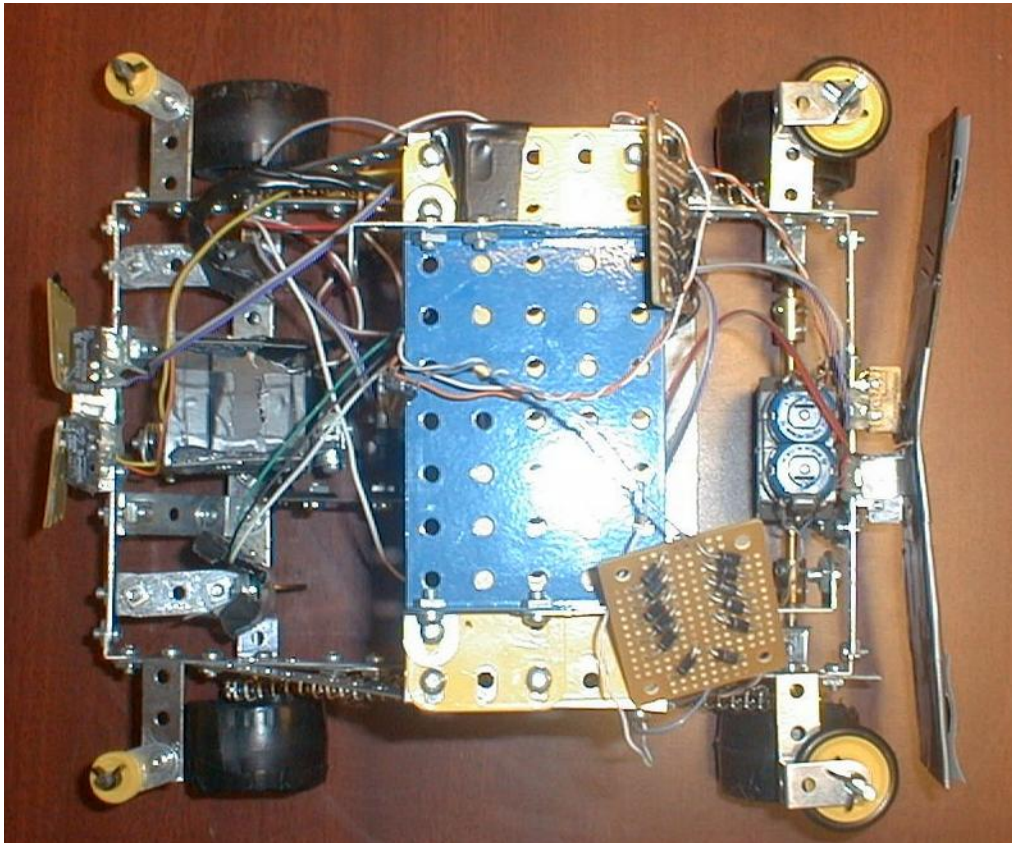
Simons & Chabris (1999) – only 50% of subjects see the ……..
Using “Count white team passes” affordance rather than complete analysis

Contact sensors

Principal function is location

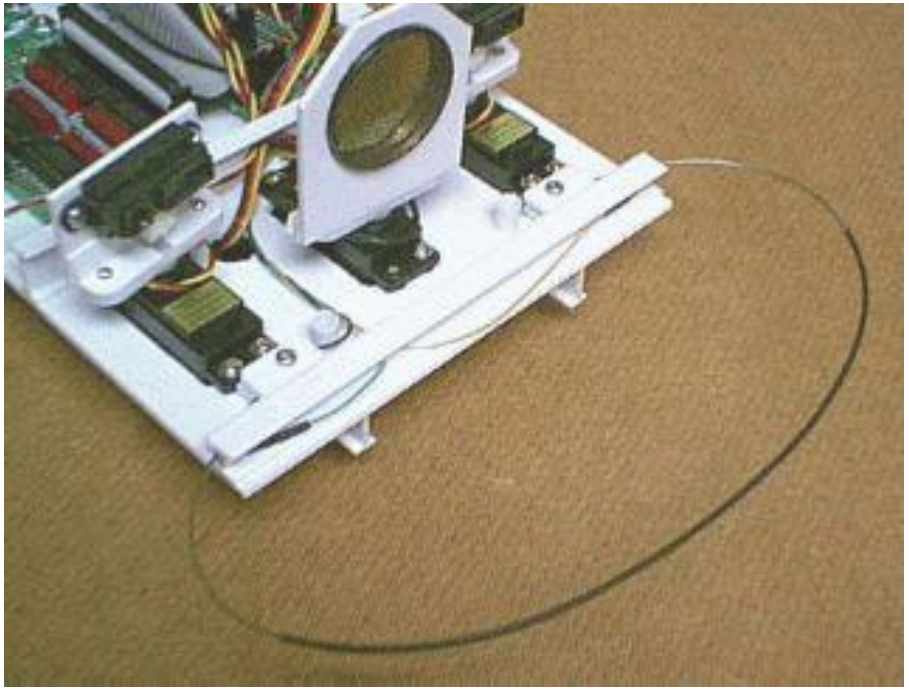
e.g. bump switch or pressure sensor:

Is the object contacting this part of the robot?



Contact sensors

Principal function is location
e.g. bump switch or pressure sensor:
Is the object contacting this part of the robot?
Antennae: Extend the range with flexible element



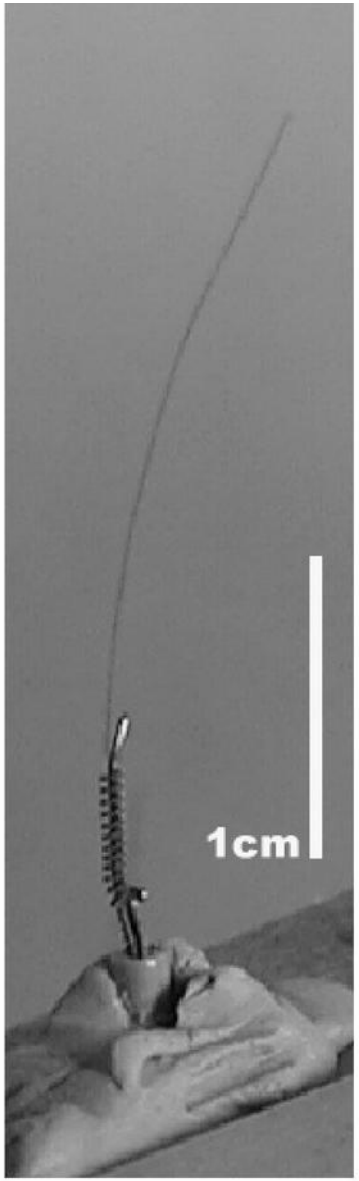
Contact sensors

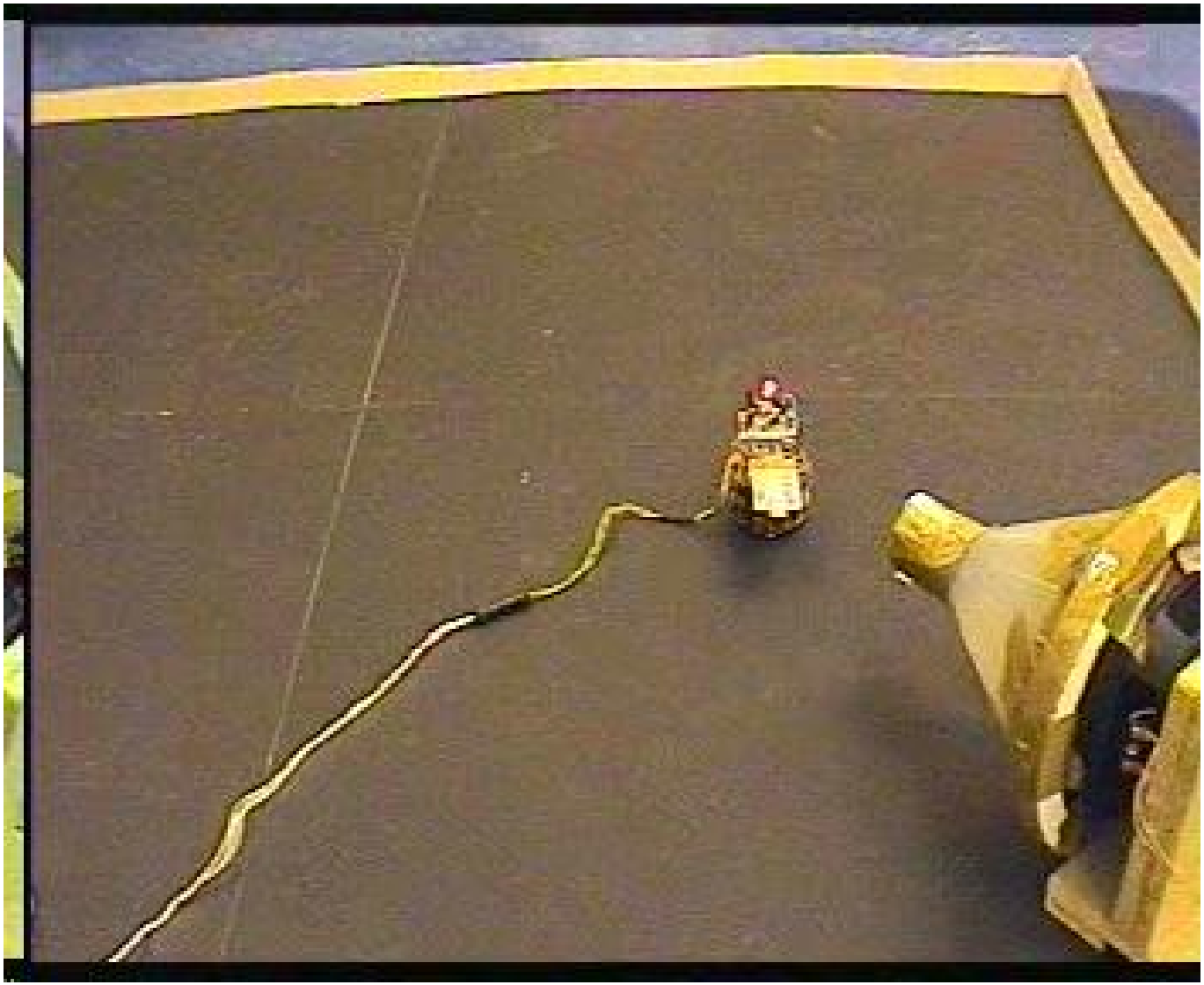
Can also use for recognition e.g.

- Is it moving or are you?
- Human touch can distinguish shape, elastic force, slip, surface texture, …
- Rat whiskers also use active sensing

‘Contact’ sensors

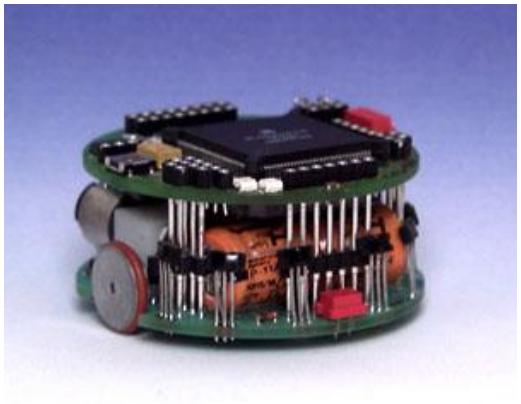
Note these kinds of sensors can also be used to detect flow e.g. wind sensors

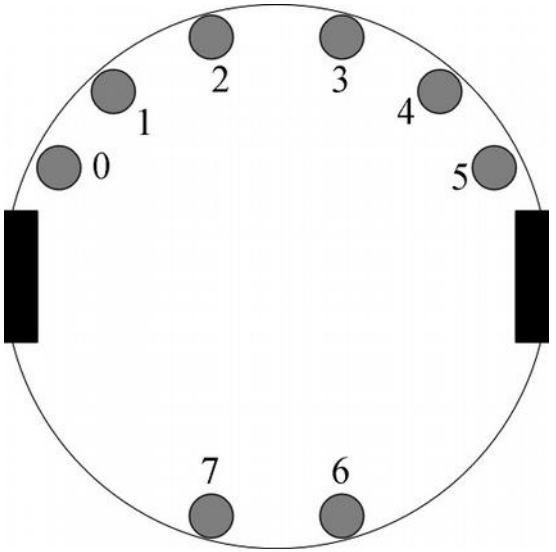




Proximity and range sensors

- Again main function is position: distance to object at specific angle to robot
- Typically works by emitting signal and detecting reflection
- Short-range: proximity sensor, e.g. IR





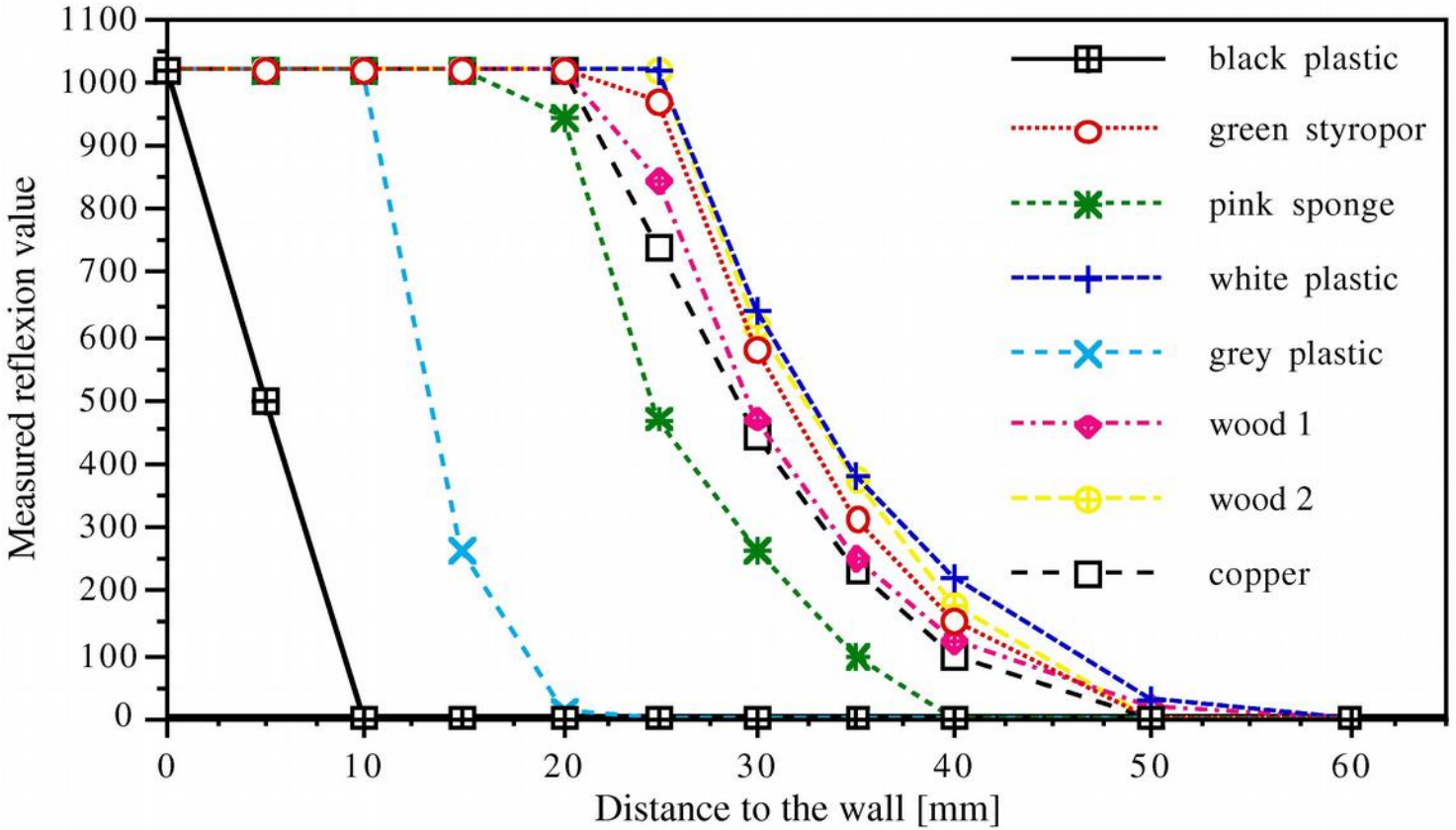
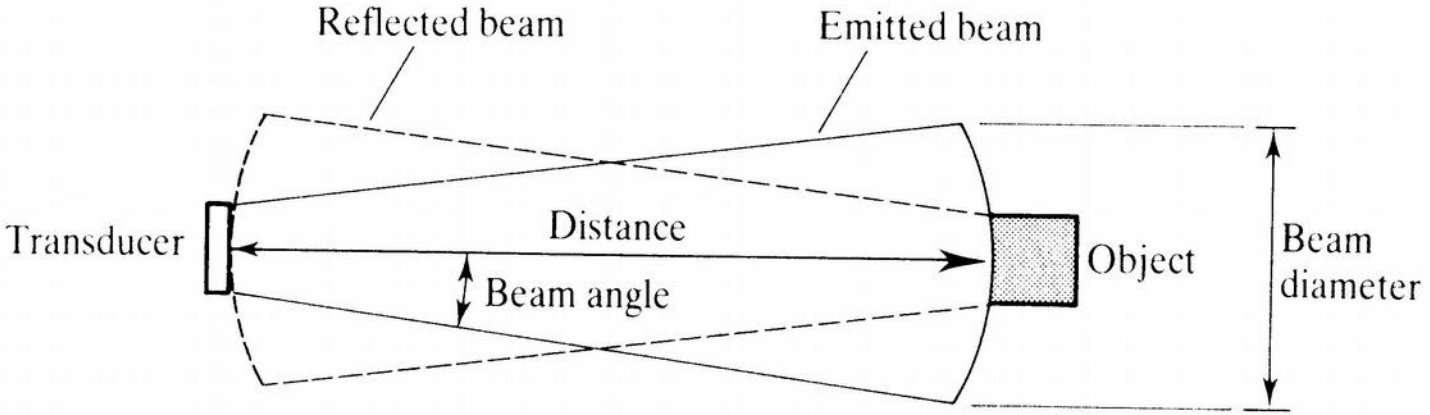


Figure 11: Measurements of the light reflected by various kinds of objects versus the distance to the object.

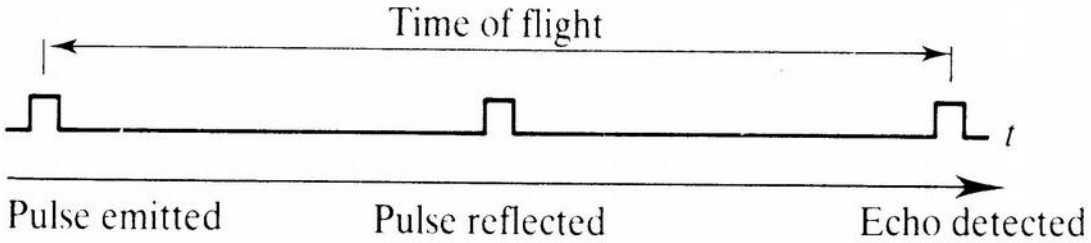
Proximity and range sensors

Over longer distance: range sensors e.g. Sonar: emit sound and detect reflection

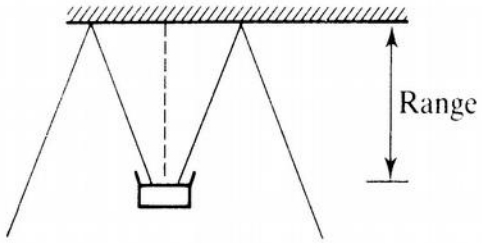


$$\text{Distance} = \frac{\text{Time of flight} \times \text{Speed in medium}}{2}$$

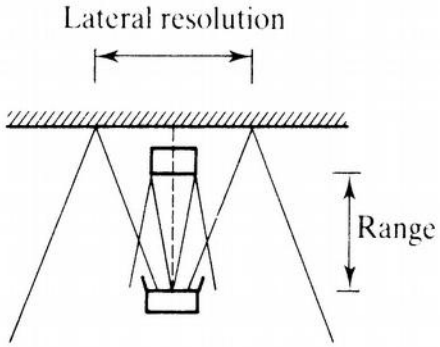
$$\text{Beam diameter} = 2 \times \text{distance} \times \tan(\text{beam angle})$$



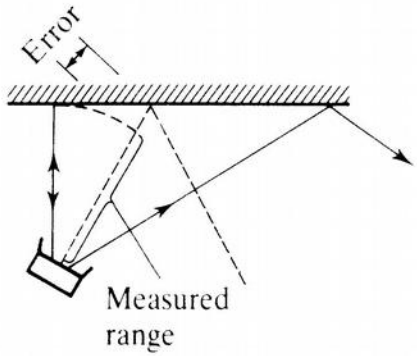
a. Sonar reflection time gives range
b. Can only resolve objects of beam width
c. Apparent range shorter than axial range
d. Angle too large so wall invisible
e. Invisible corner
f. False reflection makes apparent range greater



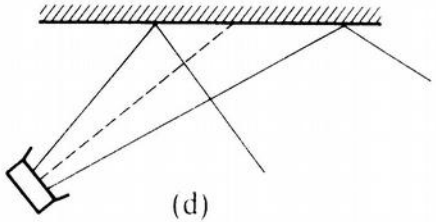
(a)



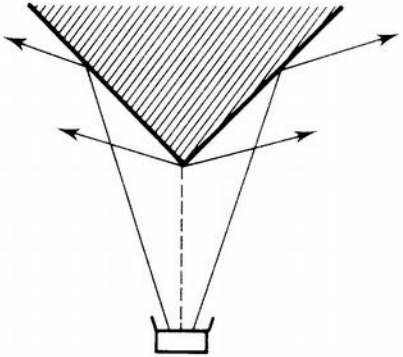
(b)



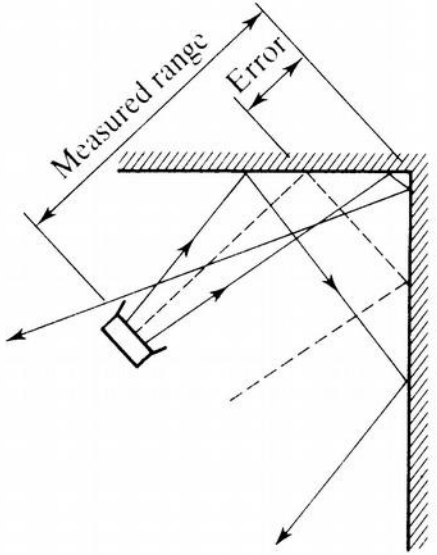
(c)



(d)



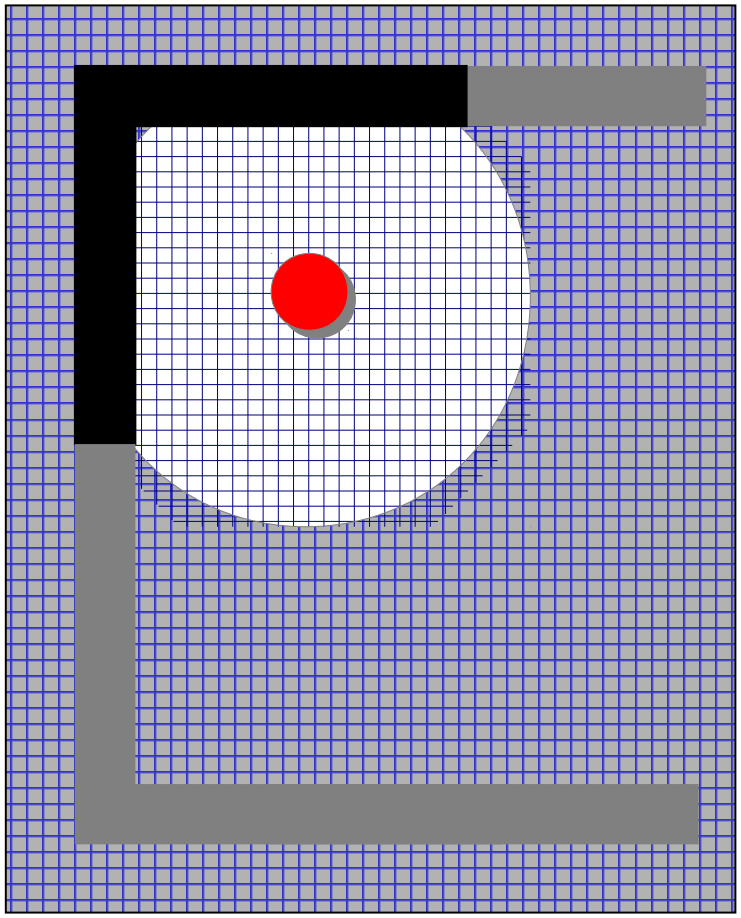
(e)



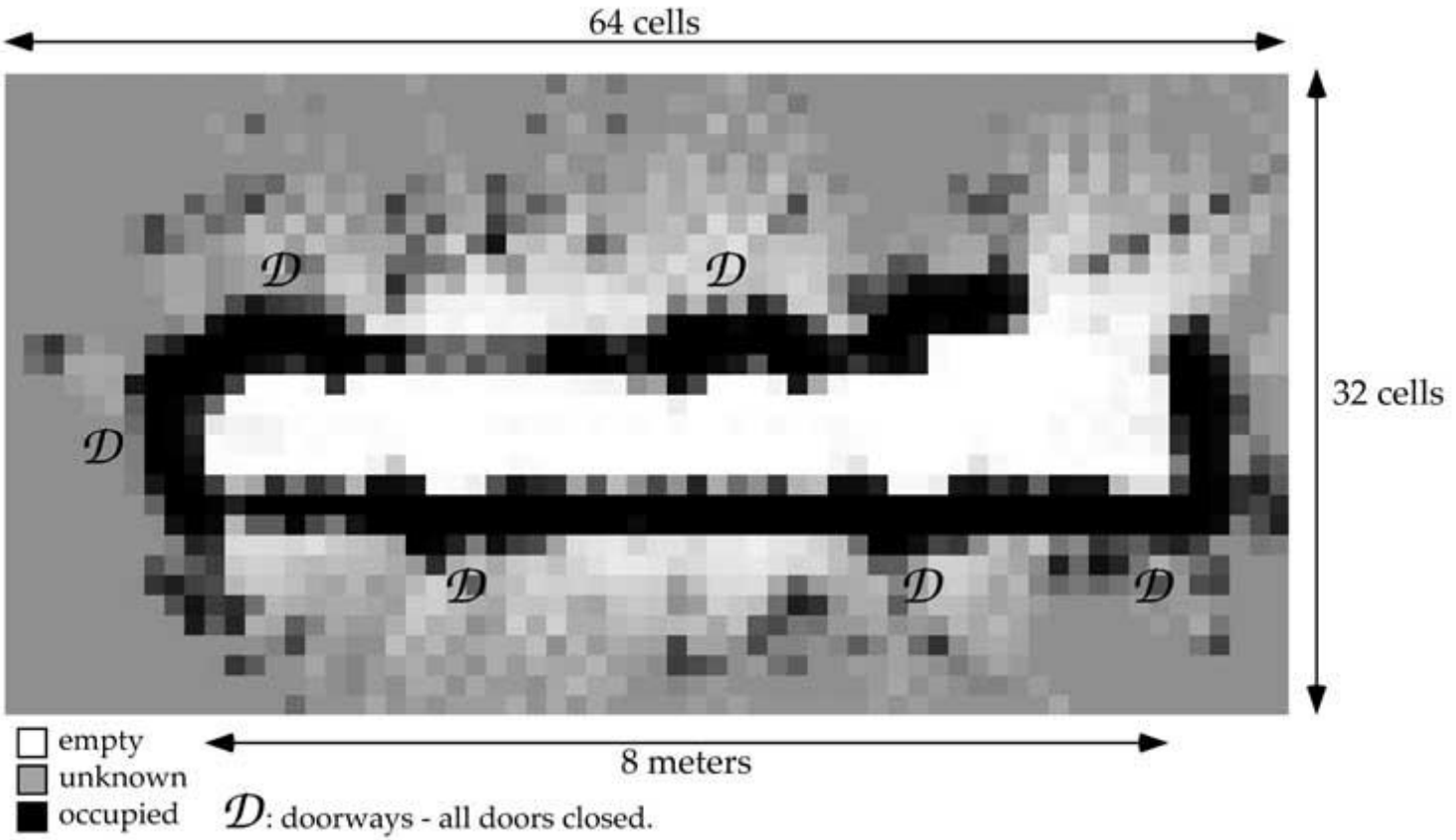
(f)

Using sonar to construct an occupancy grid

- Robot wants to know about free space
- Map space as grid
- Each element has a value which is the probability it contains an obstacle
- Update probability estimates from sonar readings



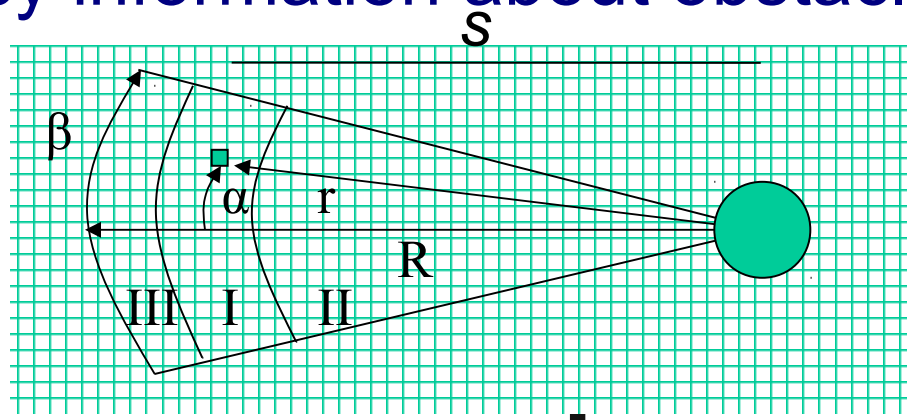
Sample occupancy grid



Noisy fusion of multiple sonar observations

Learning the map

Assuming robot knows where it is in grid, sensory input provides noisy information about obstacles, e.g. from sonar



Probability $p(z|O)$ of grid element $z=(r,\alpha)$ in region I

$$\frac{1}{Z}\exp -\frac{1}{2}\left[\left(\frac{(s-r)^2}{2(r\sigma_1)^2}\right)+\left(\frac{\alpha^2}{2(\beta\sigma_2)^2}\right)\right]$$

if occupied (O) given measurement s

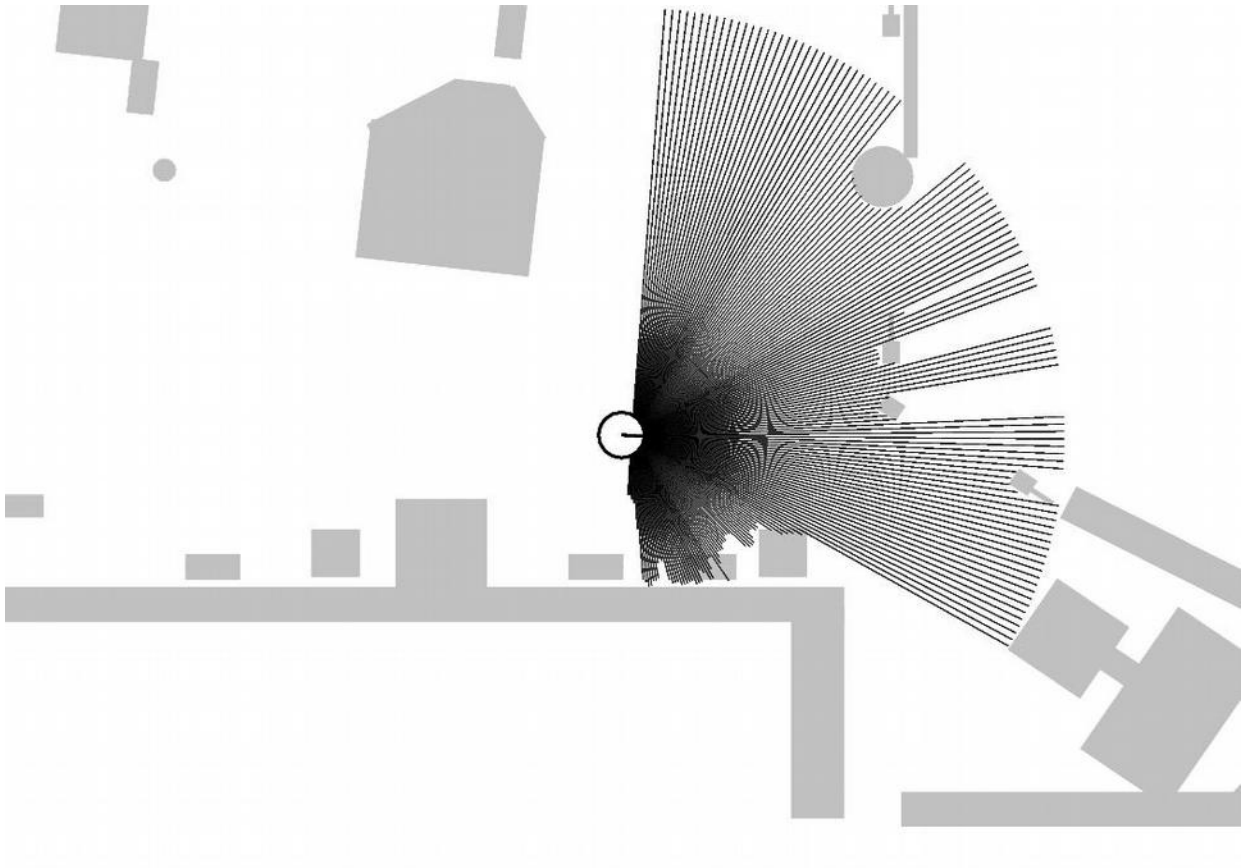
Using Bayesian approach where $p(O)$ will depend on previous measurements

$$p(O|z)=\frac{p(z|O)\,p(O)}{p(z|O)\,p(O)+p(z|\sim O)\,p(\sim O)}$$

Proximity and range sensors

- More accurate information from laser rangefinder
- Either planar or scanning
- 1,000,000 pixels per second
- Range of 100s of metres (typical values for robots)
- Accuracy of sub-mm
- Errors possible due to reflections, temperature gradients or small objects

Sample laser scan



Light sensors

- Why is it so useful to detect light?
- Straight lines mean the rays reflected from objects can be used to form an image, giving you ‘where’.
- Very short wavelengths gives detailed structural information (including reflectance properties of surface, seen as colour) to determine ‘what’.
- Very fast, it is especially useful over large distances.
- But requires half our brain to do vision…

Conclusions and outlook

- Robots need sensing: location, objects, obstacles
- Commonly used sensors: laser range, sonar, contact, GPS (outdoors), markers
- General scene scanning *vs.* affordances
- Proprioceptic sensors monitor the state of the robot (later)
- Redundant sensors: Sensor fusion
- Distributed sensing in cooperative robots

IVR 10: Effectors and Actuators

12/2/2015

Overview:

- Mechanisms for acting on the world
- Degrees of freedom and mobility
- Methods of locomotion: Wheels, legs and beyond
- Methods of manipulation: Arms, grippers
- Methods of actuation & choices of hardware
- Control problem: Mapping from signals to actuators to desired world effects

J. Michael Herrmann. michael.herrmann@ed.ac.uk. phone: 0131 6 517177, IF 1.42

What is a robot?

- A robot is an embodied artificial **agent**.
  In practice, it is usually an electro-mechanical machine which is guided by computer, and is able to perform tasks with some degree of independence from external guidance.
- Robots tend to do some or all of the following:
  1. **Sense** their environment as well as their own state
  2. Exhibit **intelligence** in behaviour, especially planned behaviour which mimics humans or other animals.
  3. **Act** upon their environment, move around, operate a mechanical limb, sense actively, communicate, ...

adapted from Wikipedia

"Acting and sensing are still the hardest parts."

(D. Kortenkamp, R. P. Bonasso oder R. Murphy)

Knowledge component

- Computer or brain-like processing device, (symbolic/subsymbolic/hybrid)
- Preprocessing of sensory signals
- Memory: semantic, episodic, declarative, logical
- Adaptation rules for the knowledge components
- Strategy, planning and evaluation
- Working memory
- Actuator control

Replicating fossil paths with toilet roll

T. Prescott & C. Ibbotson (1997)

Control combines thigmotaxis (stay near previous tracks) & phobotaxis (avoid crossing previous tracks)

Components of robots

- Sensory components: Acquisition of information
- Information processing and control
- Actuatory and effectory components: Realization of actions and behaviour

- Plus: Batteries, communication devices, interfaces, central executive, self-evaluation, learning mechanisms, tools for inspection, middleware, ...

Sensor categories

- Exteroception: Perception of external stimuli or objects (distance and contact)
- Proprioception: Perception of self-movement and internal states
- Exproprioception: Perception of relations and changes of relations between the body and the environment (combination of proprioception and exteroception)

Actuatory components

Actuators and effectors can be classified into
(analogous to the sensory part?)

- components relating to the environment
- components relating to the own body
- components relating to perception
- components relating to communication

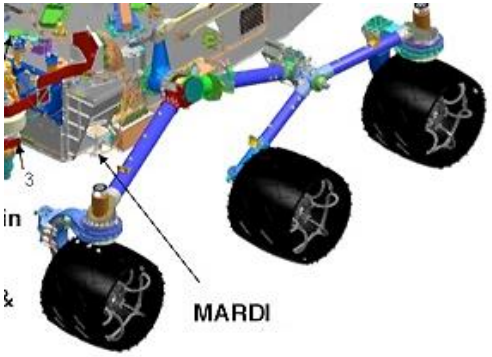
Effectors and Actuators


John Amend / Cornell University

Key points:

- Mechanisms for acting on the environment
- ‘Degrees of freedom’
- Methods of locomotion: Wheels, legs etc.
- Methods of manipulation: Arms, grippers, suction cups
- Methods of actuation and transmission
- The control problem: Relations between input signals and actuators and the desired effects

Effector: A device that affects the physical environment

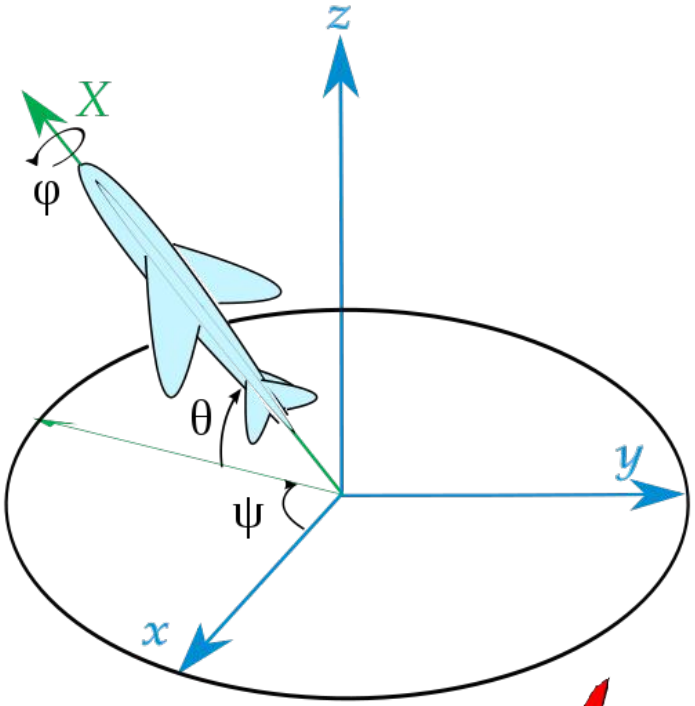


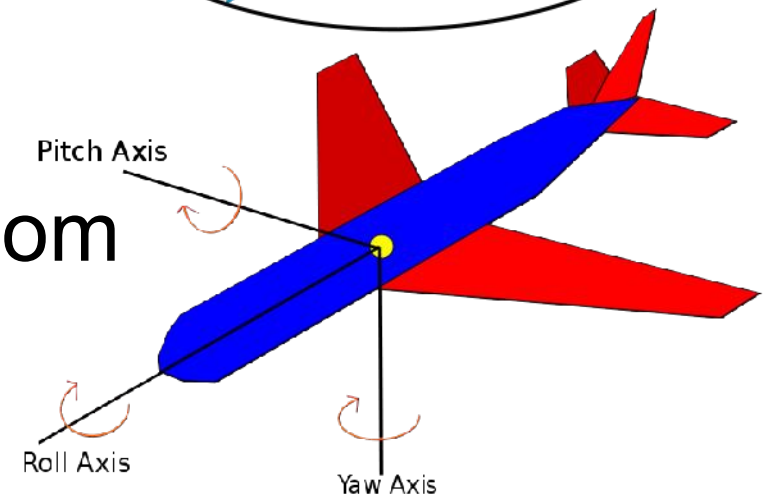
- Choice of effectors sets upper limit on what the robot can do
- Locomotion:
  - Wheels on a mobile robot
  - or legs, wings, fins, claws, suction cups …
  - Whole body movements: Snakes, caterpillars
- Manipulation:
  - Grippers on an assembly robot
  - or welding gun, paint sprayer, …
  - Whole body might be used push objects

- In both cases consider the *degrees of freedom* in the design
- Further option: Effects by signals such as speakers, light, pen

Degrees of freedom

- General meaning: How many parameters needed to describe a rigid object?
- E.g. for an object in space:
  - Position: *x*,*y*,*z*
  - Rotation: Roll, pitch, yaw





- Total of 6 degrees of freedom
- How many d.o.f. to specify a vehicle on a flat plane?

Odometry

Odometry: Position measurement by distance travelled

- Know current position (x, y, θ)
- Know how much wheels rotate
- New position =

  old position + commanded motion

Note:

- Errors add up → good calibration required
- Motors/timing may be inaccurate

  → use shaft encoders
- Wheels slip on surface → use landmarks/beacons need some feature tracking

Degrees of freedom

In relation to robots consider:

- How many joints/articulations/moving parts?
- How many individually controllable moving parts?
- How many independent movements with respect to a co-ordinate frame?
- How many parameters to describe the position of the whole robot or its end effector?

- How many moving parts?
  - If parts are linked need fewer parameters to specify them.
- How many individually controllable moving parts?
  - Need that many parameters to specify robot's configuration.
  - Often described as 'controllable degrees of freedom'
  - But note, robot may be *redundant* e.g. two movements may be in the same axis
  - Alternatively called 'degrees of mobility'

- How many degrees of mobility in a human arm? In a knee joint?
- How many degrees of mobility in the arm of an octopus?

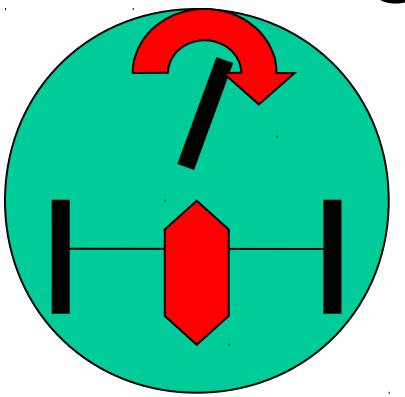
- Redundant manipulator

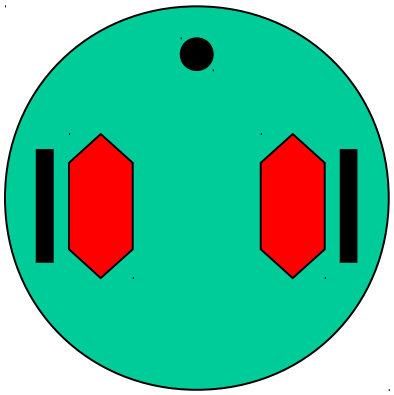
  Degrees of mobility > degrees of freedom
- Result is that have more than one way to get the end effector to a specific position

- How many independent movements with respect to a co-ordinate frame?
  - Controlled degrees of freedom of the robot
  - May be less than controllable degrees of freedom
- How many parameters to describe the position of the whole robot or its end effector?
  - For fixed robot, d.o.f. of end effector is determined by d.o.f. of robot (max 6)
  - Mobile robot on plane can reach position described by 3 d.o.f., but if the robot has fewer d.o.f. then it cannot do it *directly* – it is *non-holonomic*

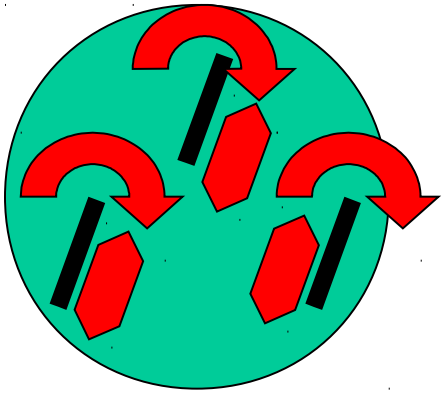
Alternative vehicle designs

- ‘Car’- steer and drive



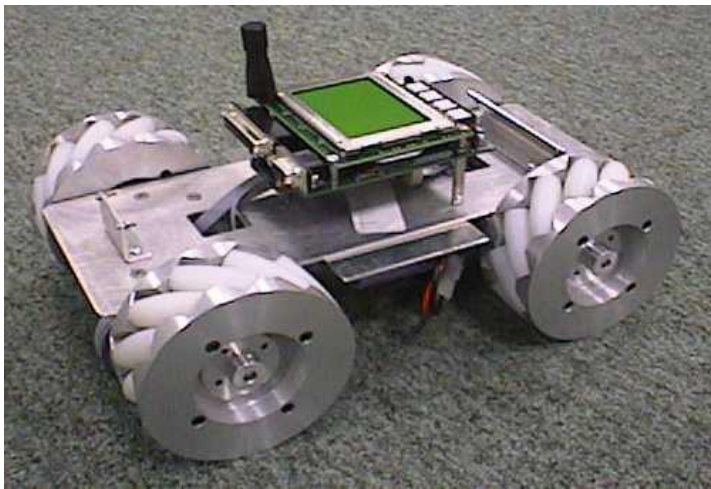


- Two drive wheels and castor 2DoF



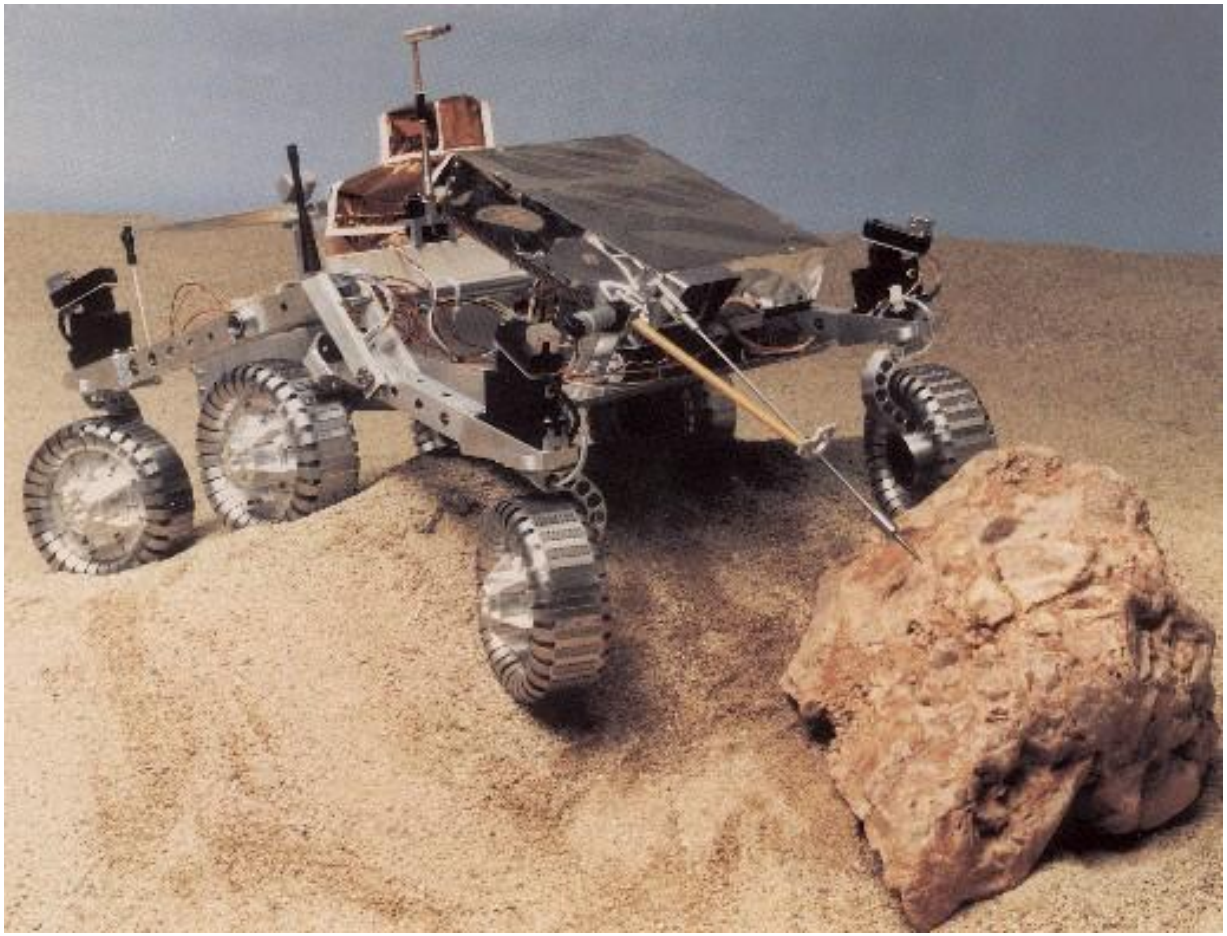
- Three wheels that both steer and drive

- Omni-wheels may be easier for path planning, but is mechanically more complex



Locomotion on uneven terrain

- Use the world (ramps etc.)
- Larger wheels
- Suspension
- Tracks



Alternative is to use legs

Note: Wheels and variants are faster, for less energy, and are usually simpler to control

Legged locomotion

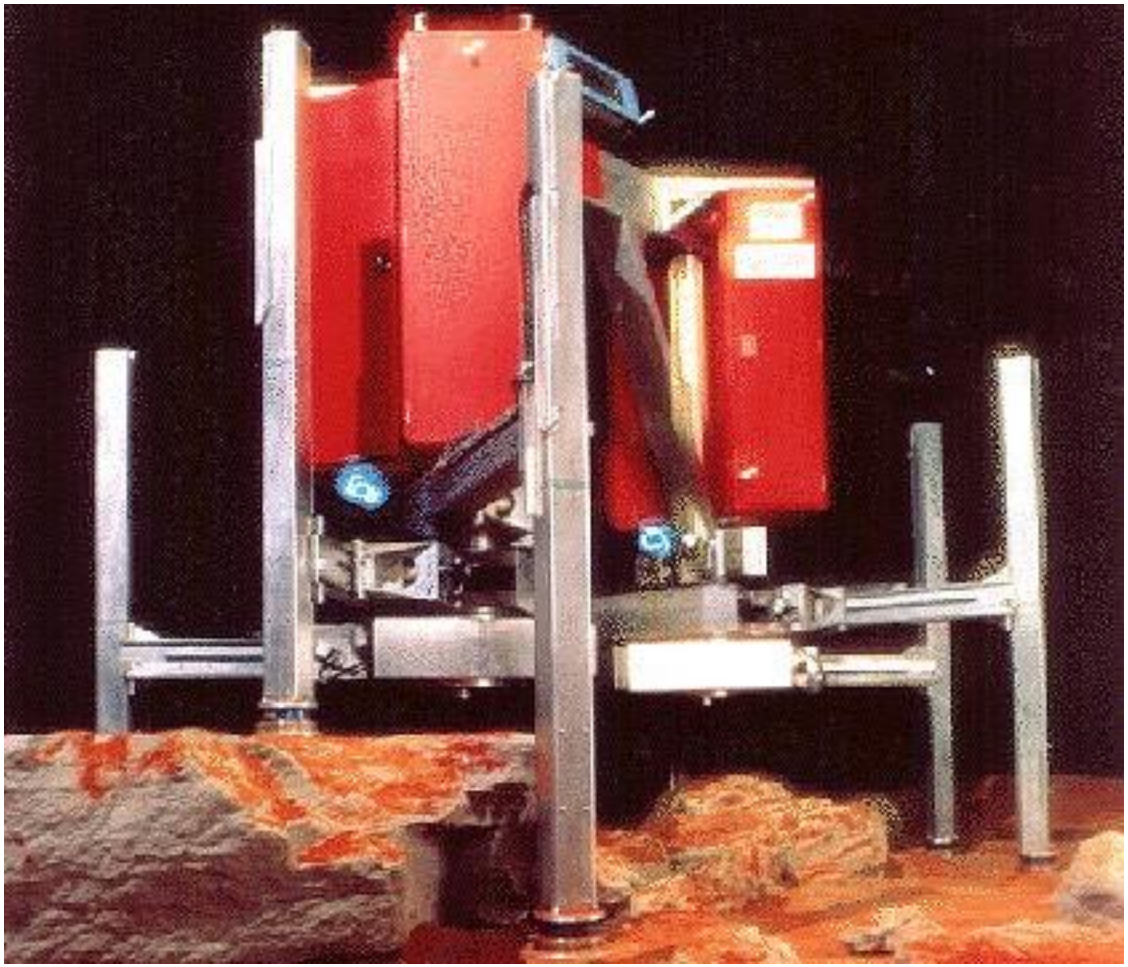
Strategies:

Statically stable control

e.g. ‘Ambler’

Whittaker, CMU

Keep three legs on ground at all times

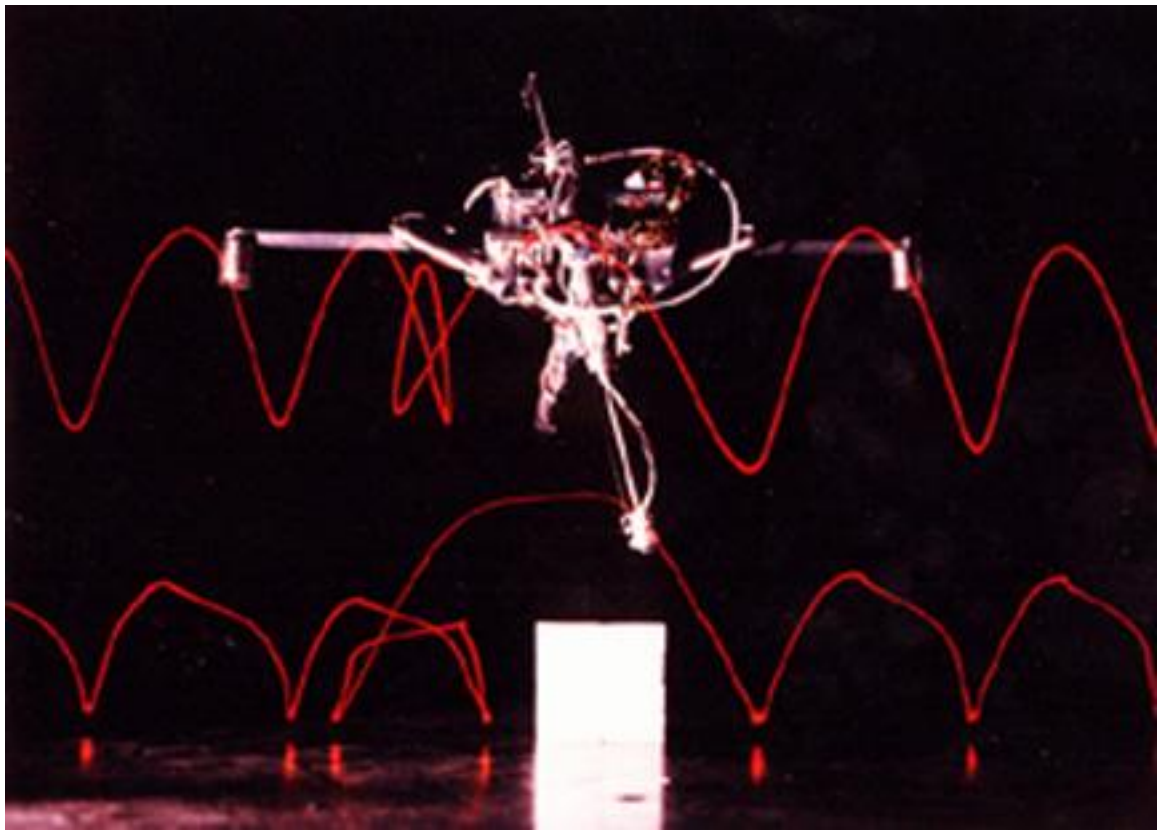


Legged locomotion

Strategies:

Dynamic balance e.g. Raibert's hopping robots

Keep motion of center of gravity within control range



Legged locomotion

Strategies:

‘Zero moment point’ control, e.g. ASIMO

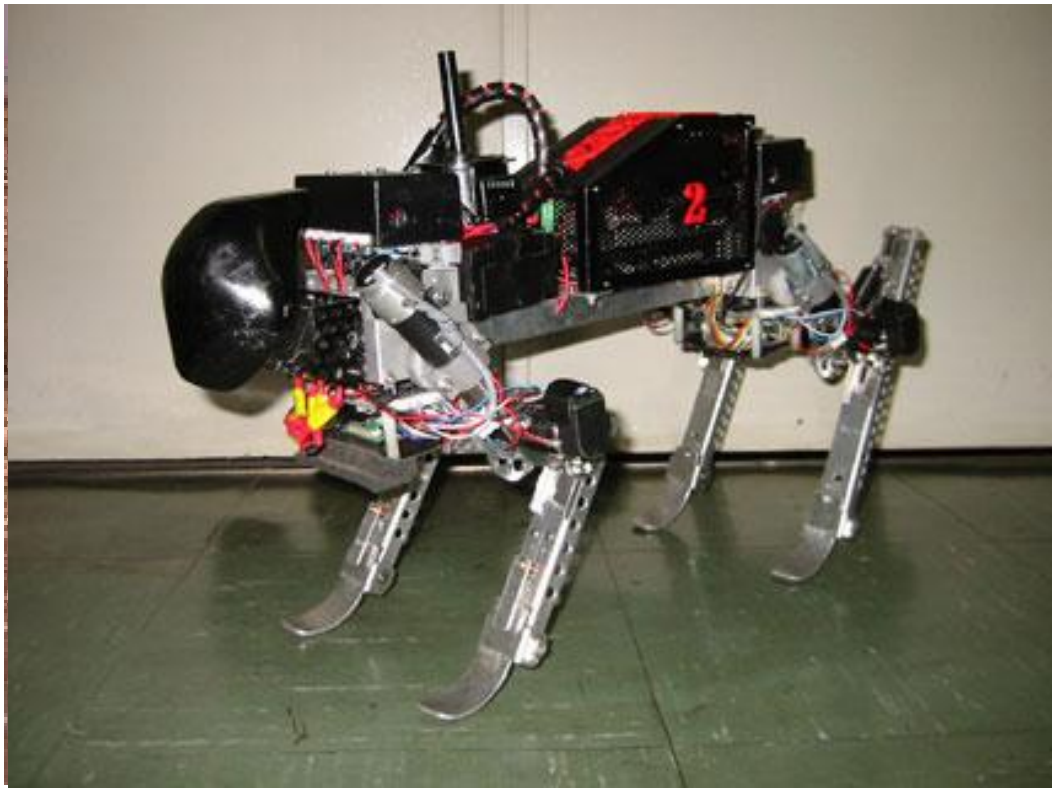
Keep point where static moment is zero within foot contact hull

Legged locomotion

Strategies:

Limit cycle in dynamic phase space e.g. ‘Tekken’ (H. Kimura)

Cycle in joint phase space + forces that return to cycle



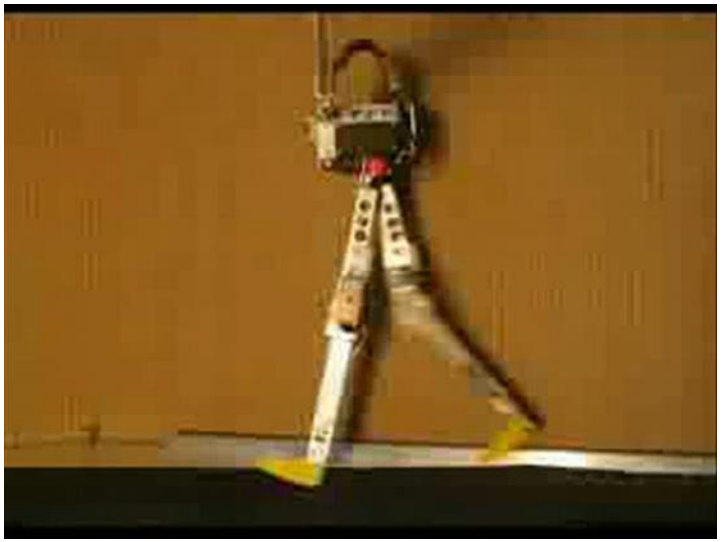
Legged locomotion

Strategy:

Exploit natural dynamics with only gravity as the actuator

E.g. passive dynamics walkers

hybrid active/passive walkers



P Manoonpong, T Geng, B Porr, F Wörgötter (2005) RunBot

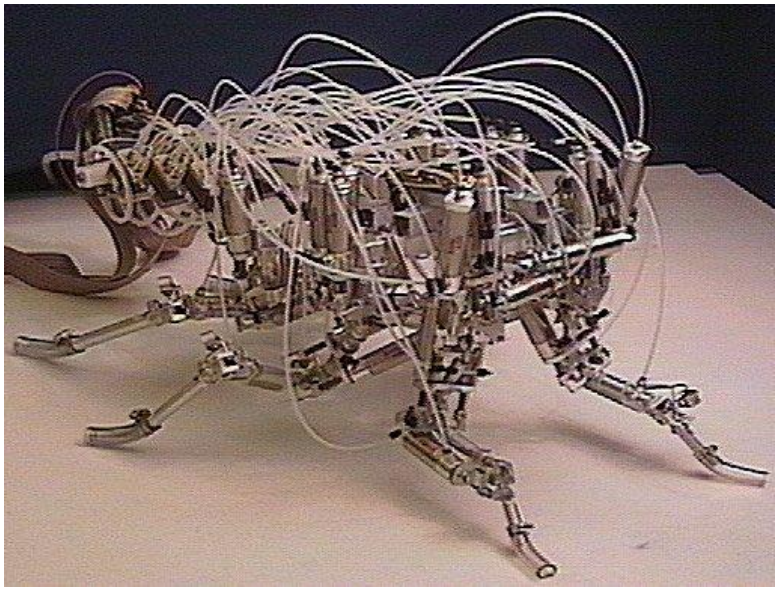
BigDog

Sensors for joint position and ground contact, laser gyroscope and a stereo vision system.

Boston Dynamics with Foster-Miller, NASA Jet Propulsion Laboratory, Harvard University Concord Field Station (2005)

E.g. Robot III vs. Whegs

Roger Quinn et al. –
biorobots.cwru.edu

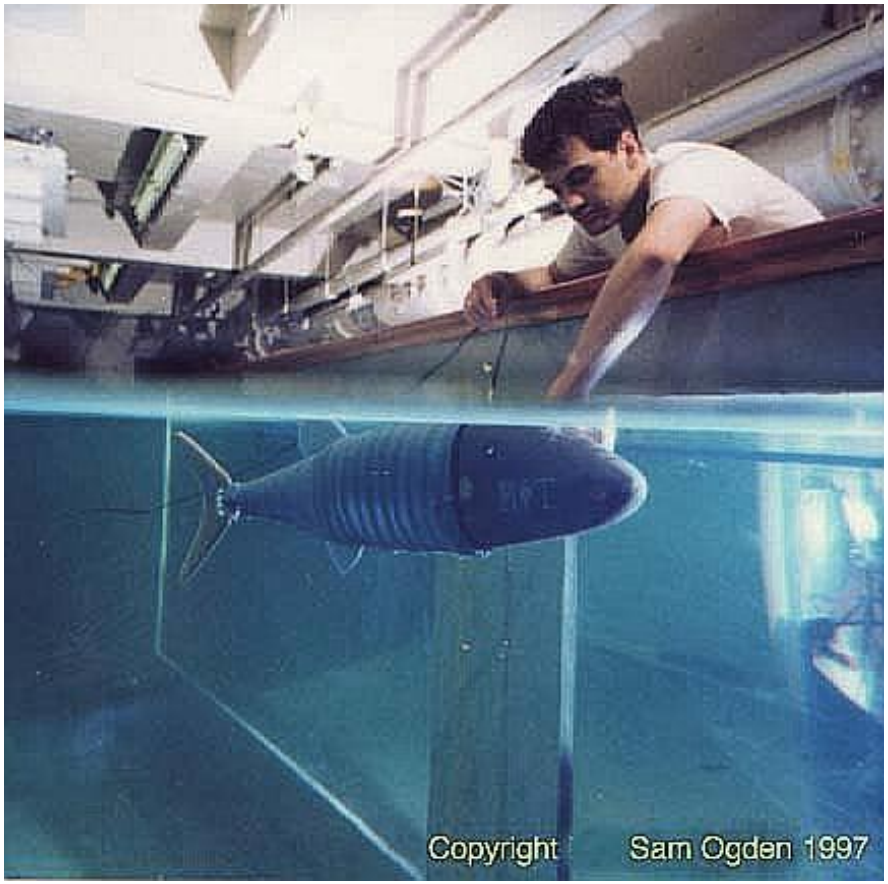


Realistic cockroach mechanics but difficult to control (Robot III),
vs. pragmatic (cricket?) kinematics, easily controllable

Exploit dynamics of mechanical system, e.g. RHex
Springiness restores object to desired state

Other forms of locomotion?

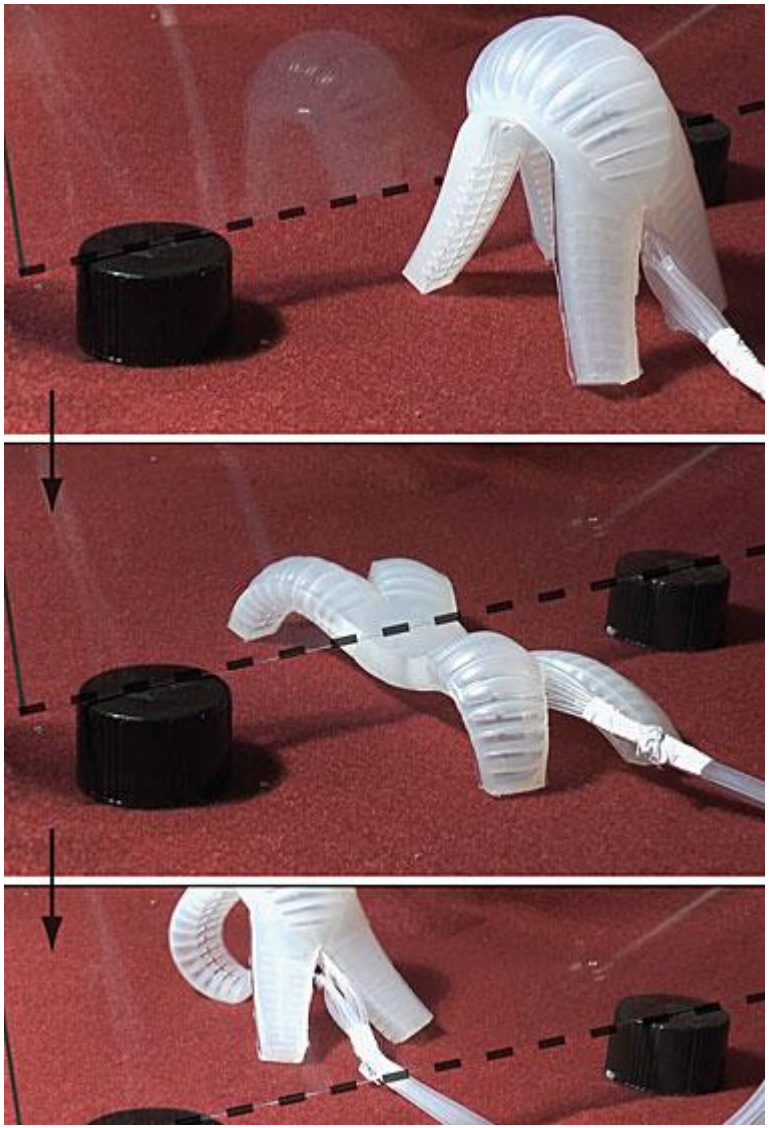
Swimming: e.g. robopike project at MIT



Flight: e.g. Micromechanical Flying Insect project at Berkeley

Other forms of locomotion?

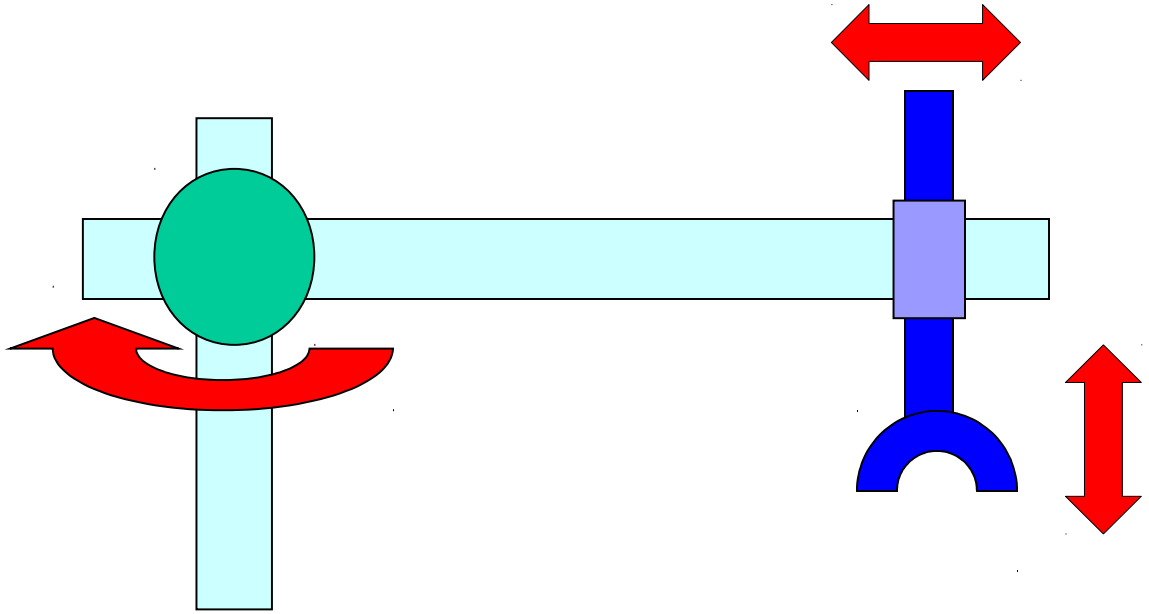
Gavin Miller:
snake robots



George M. Whitesides:
Soft robots

Robot arms

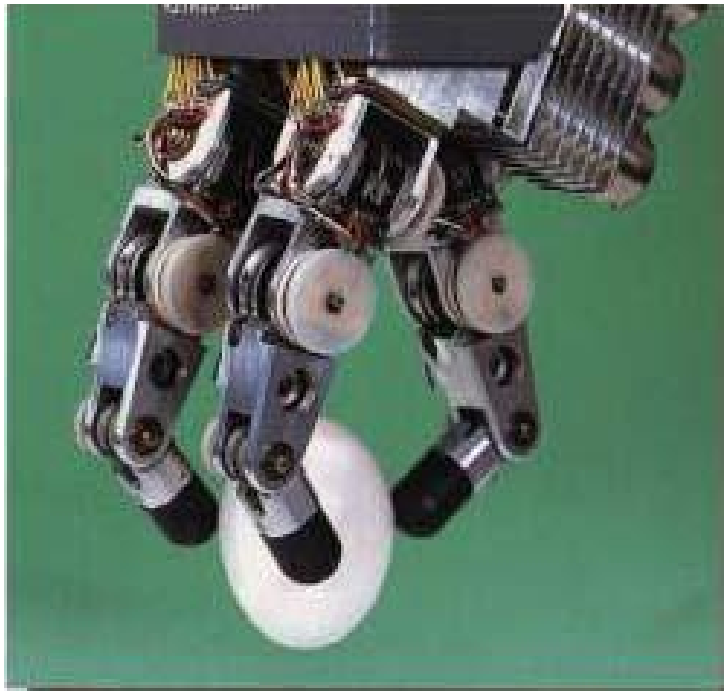
- Typically constructed with rigid *links* between movable one d.o.f. *joints*
- Joints typically
  - *rotary* (revolute) or prismatic (linear)

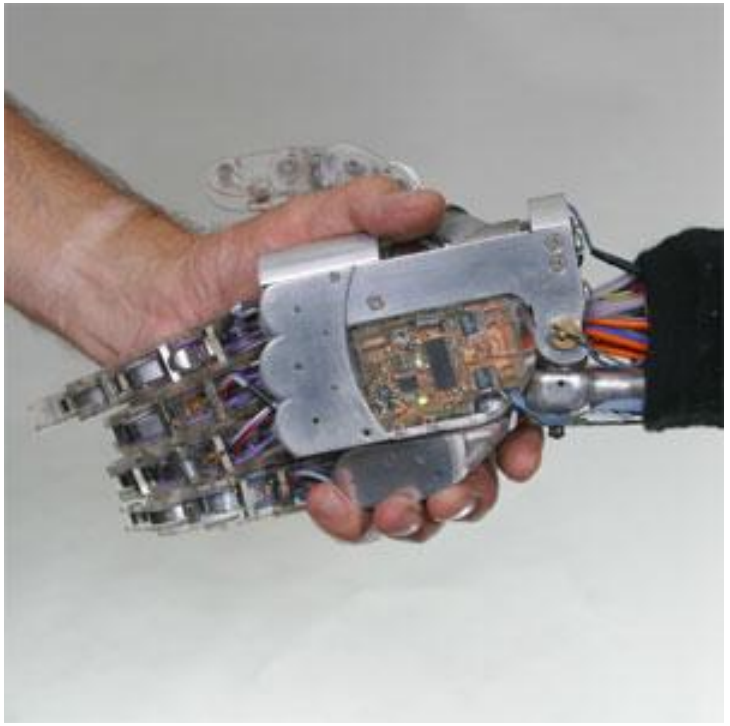


Robot arms

Robot arm end effectors

- Simple push or sweep
- Gripper – different shape, size or strength
- Vacuum cup, scoop, hook, magnetic
- Tools for specific purposes (drills, welding torch, spray head, scalpel,…)
- Hand for variety of purposes





Actuation

What produces the forces to move the effectors?

Electrical:

- DC motors (speed proportional to voltage – voltage varied by pulse width modulation)
- Stepper motors (fixed move per pulse)

Pressurised -

- Liquid: Hydraulics
- Air: Pneumatics, air muscles

Connected via transmission: system gears, brakes, valves, locks, springs…

Non-conventional actuators

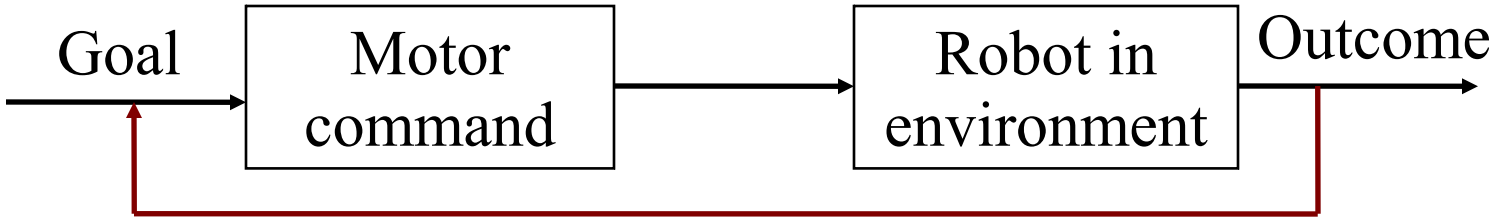
Rotary electric motors
Hydraulics
Pneumatics
Piezoelectric
Bimetallic
Shape memory alloy

Electromechanical
Pistons
Electrochemical
Magnetostrictiver
Rheological
...

Issues in choosing actuators

- Load (e.g. torque to overcome own inertia)
- Speed (fast enough but not too fast)
- Accuracy (will it move to where you want?)
- Resolution (can you specify exactly where?)
- Repeatability (will it do this every time?)
- Reliability (mean time between failures)
- Power consumption (how to feed it)
- Energy supply & its weight
- Usually many possible trade-offs between physical design and ability to *control*

The control problem



- For given motor commands, what is the outcome? *= Forward model*
- For a desired outcome, what are the motor commands? *= Inverse model*
- From observing the outcome, how should we adjust the motor commands to achieve a goal? *= Feedback control*

The control problem

Want to move robot hand through set of positions in task space – $X(t)$

$X(t)$ depends on joint angles in the arm $A(t)$

$A(t)$ depends on the coupling forces $C(t)$ delivered by the transmission from the motor torques $T(t)$

$T(t)$ produced by the input voltages $V(t)$

$$V(t) \longleftrightarrow T(t) \longleftrightarrow C(t) \longleftrightarrow A(t) \longleftrightarrow X(t)$$

The control problem

$$V(t) \longleftrightarrow T(t) \longleftrightarrow C(t) \longleftrightarrow A(t) \longleftrightarrow X(t)$$

- Depends on geometry & kinematics: can mathematically describe the relationship between motions of motors and end effector as transformation of co-ordinates
- as well as on dynamics: motion depends on forces, such as inertia, friction, etc
- **Forward kinematics** is hard but usually possible
- **Forward dynamics** is very hard and at best will be approximate
- But what we actually need is ***backwards kinematics and dynamics***

Summary

- Various energy sources: electrical, hydraulic, air, muscles, …
- A variety of effectors: wheels, legs, tracks, fingers, tools, …
- Degrees of freedom and joints
- Calculating control may be hard: Choose either a sufficiently simple environment or adapt to the environment by learning

Bio-inspired Robotics

Valentino Braitenberg (1986) Vehicles: Experiments in Synthetic Psychology. MIT Press.

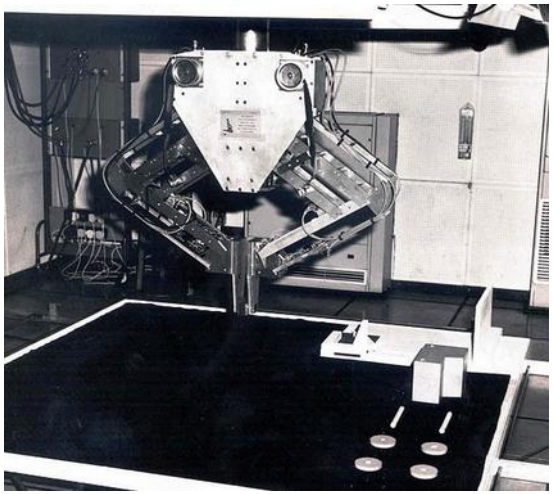
Simulation of Braitenberg Vehicles:
http://people.cs.uchicago.edu/~wiseman/vehicles/

Rodney Brooks (1999) Cambrian Intelligence: The Early History of the New AI. MIT Press.
Rolf Pfeifer & Josh Bongard (2006) How the Body Shapes the Way We Think: A New View of Intelligence. MIT Press.

Lecture **11**: (23/02/15)

Reaching and Grasping

- Reference Frames
- Configuration space
- Reaching
- Grasping

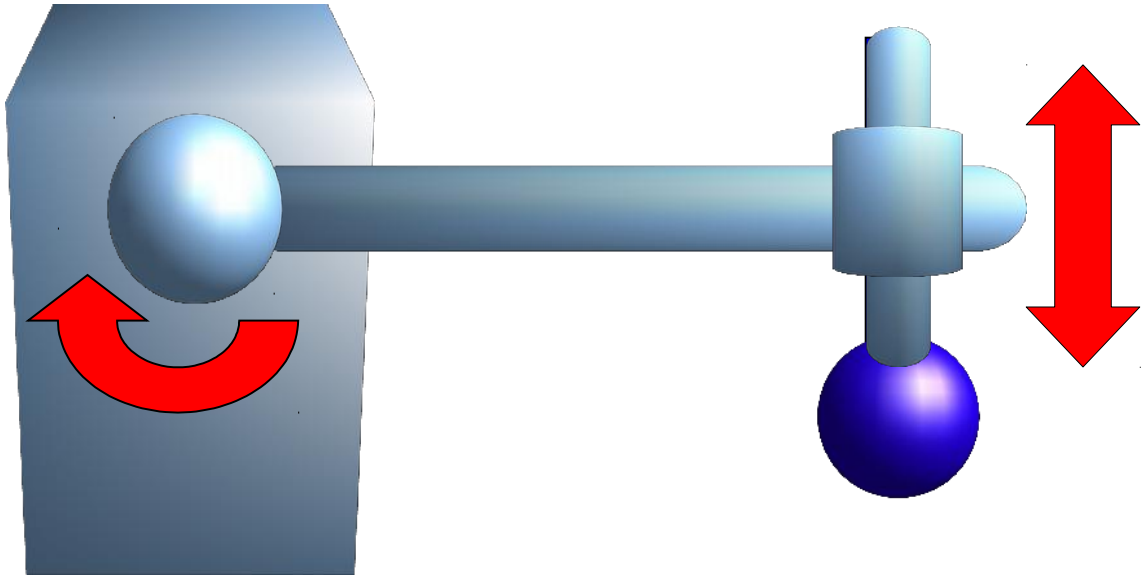


Michael Herrmann

michael.herrmann@ed.ac.uk, phone: 0131 6 517177, Informatics Forum 1.42

Robot arms

- Typically constructed with rigid *links* between movable one d.o.f. *joints*
- Joints typically
  - *rotary* (revolute) or *prismatic* (linear)



Kinematic pairs

Lower pairs

- Rotary (hinge, 1 axis)
- Prismatic (linear)
- Cylindrical (rotary + linear)
- Spherical (3 rotary)
- Planar (2 linear + rotary)

Higher pairs (different curvature): rolling bearing, cam joint, …

Wrapping pair: belt, chain, …

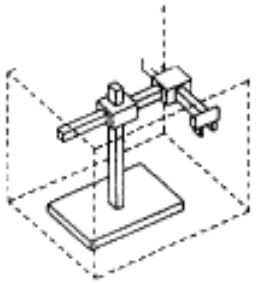
⇒ Linkages: Crankshaft-piston, …

⇒ Kinematic chains

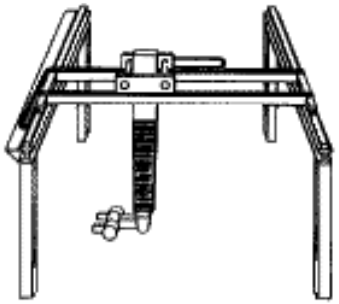
Robot arms
(manipulators)

Classification by geometry

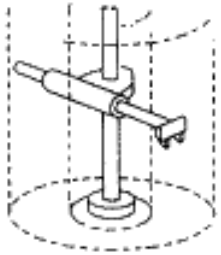
Variants



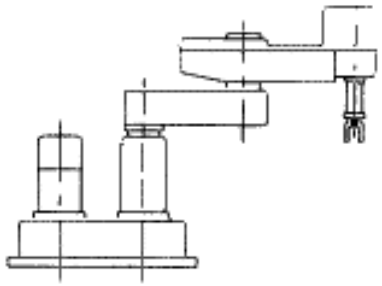
Rectangular Coordinate Robot



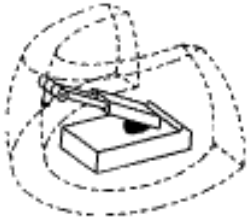
Gantry Robot



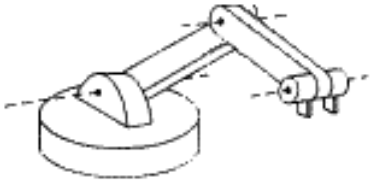
Cylindrical Coordinate Robot



SCARA Robot



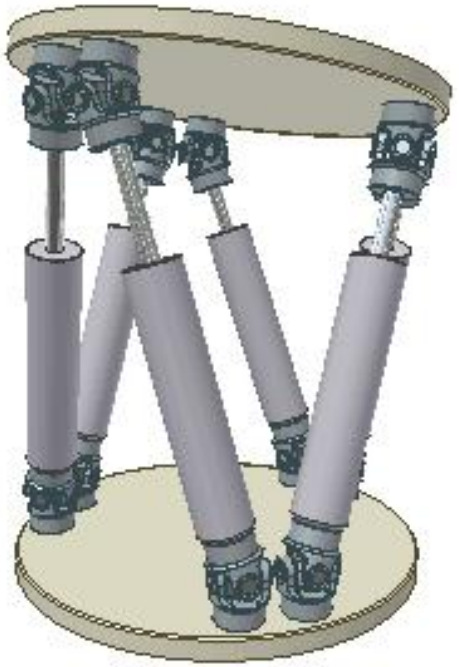
Sperical Coordinate Robot



Articulated Arm Robot

Classification by topology (kinematic chain):

- (left) Serial chain
- Parallel manipulator (below)



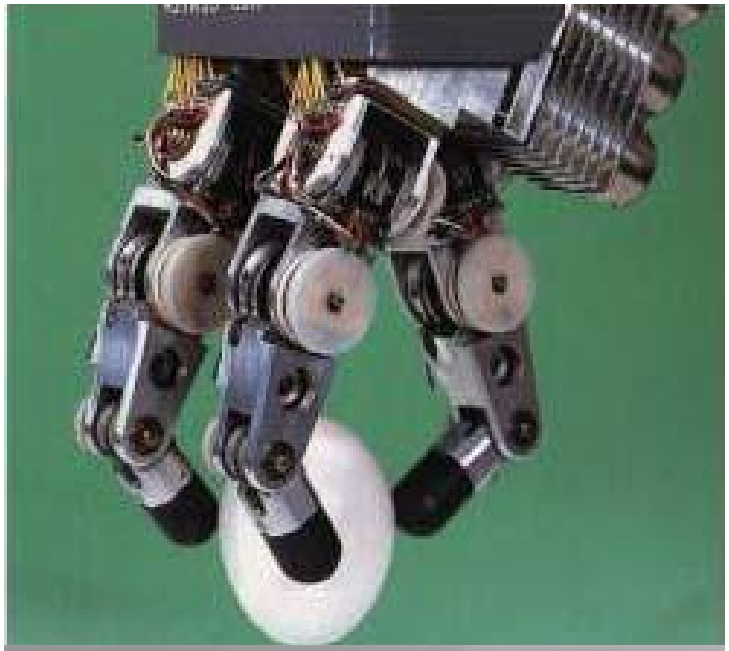
adapted from http://www.osha.gov/dts/osta/otm/otm_iv/otm_iv_4.html

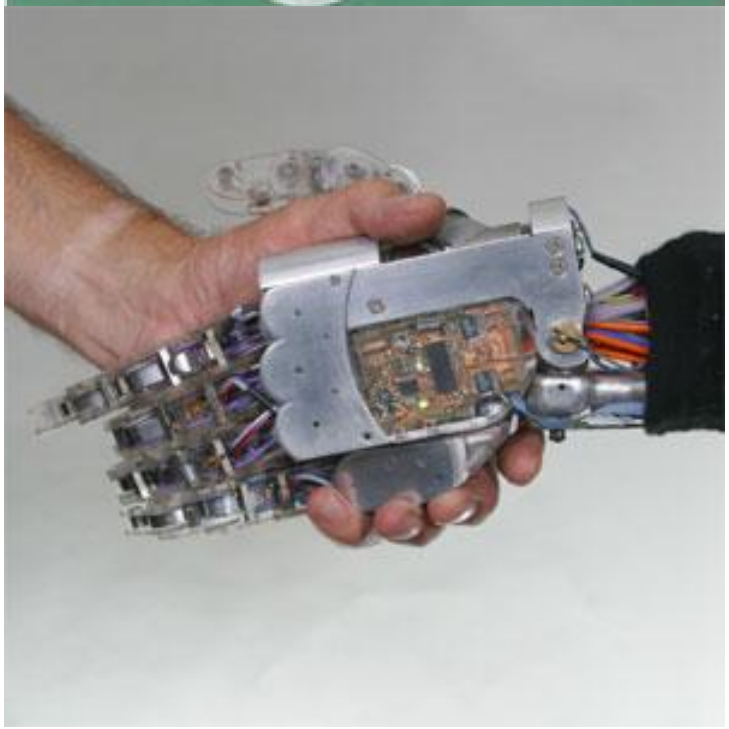
fr.wikipedia.org

Robot arm: End effectors

(at Tool Center Point at top of kinematic chain)

- Simple push or sweep
- Gripper – different shape, size or strength
- Hook, pins, needles (e.g. for handling textiles)
- Vacuum cup, magnetic
- Adhesion by contact (e.g. glue)
- Tools for specific purposes (drills, welding torch, spray head, scalpel, scoop,…)
- Hand for variety of purposes





Issues in choosing actuators

- Load (e.g. torque to overcome inertia of arm)
- Speed (fast enough but not too fast)
- Accuracy (will it move to where you want?)
- Resolution (can you specify exactly where?)
- Repeatability (will it do this every time?)
- Reliability (mean time between failures)
- Compliance (how does stiffness change?)
- Power consumption (how to feed it)
- Energy supply & its weight
- Geometry (linear vs. rotary) and other trade-offs between physical design and ability to *control*

Prehension

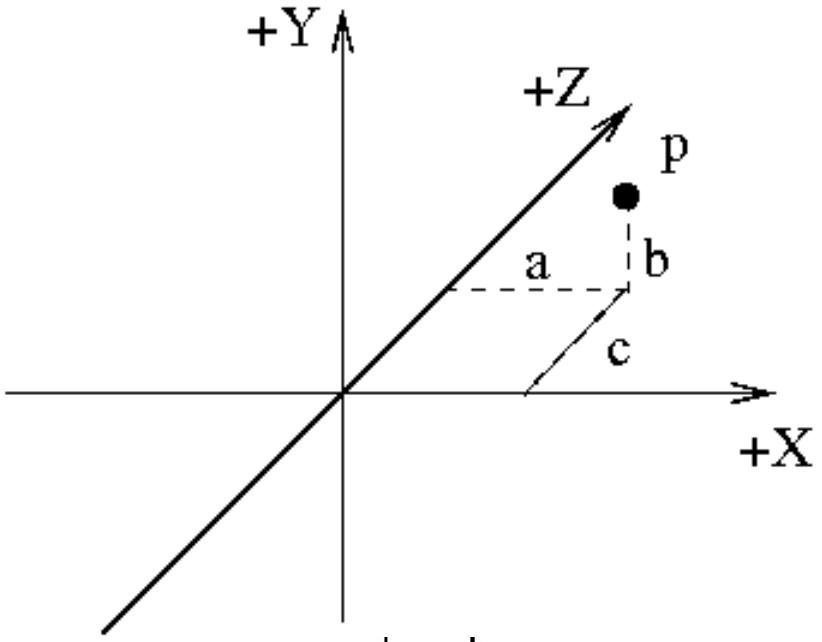
Goal: Understand ideal robot mechanisms for reaching, grasping and manipulation

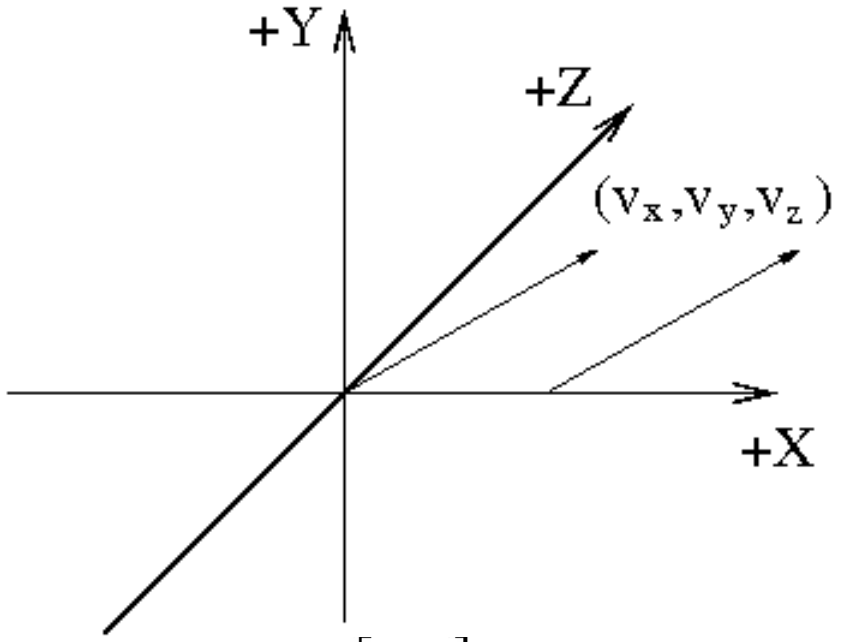
Robot positions and configurations as a function of control parameters – **kinematics**

Need to know:

- Representing mechanism geometry
- Standard configurations
- Degrees of freedom
- Grippers and graspability conditions

Vectors & Points in 3D





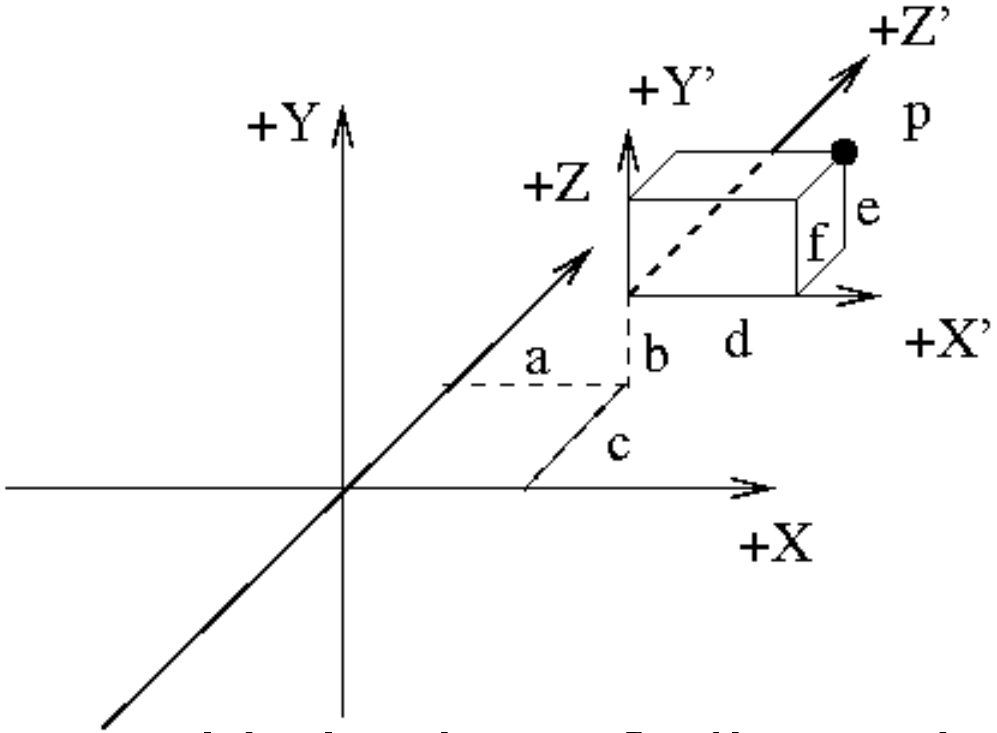
Point $\vec{p} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} = (a, b, c)^T$ Vector $\vec{v} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = (v_x, v_y, v_z)^T$

3D Coordinate Systems usually defined as left handed (right handed reverses the +*Z* direction)

Local Reference Frames

$\vec{p}$ can be expressed by

- Local (translated) coordinates (d, e, f)
- Global (untranslated) coordinates $(a+d, b+e, c+f)$

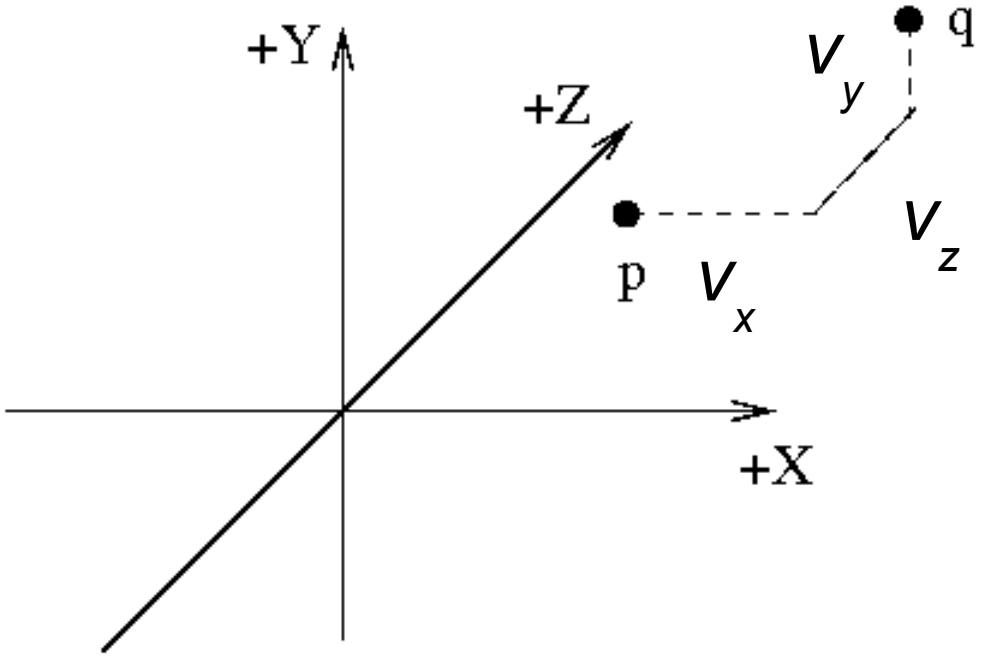


Local frame could also have further sub-local frames

Translations

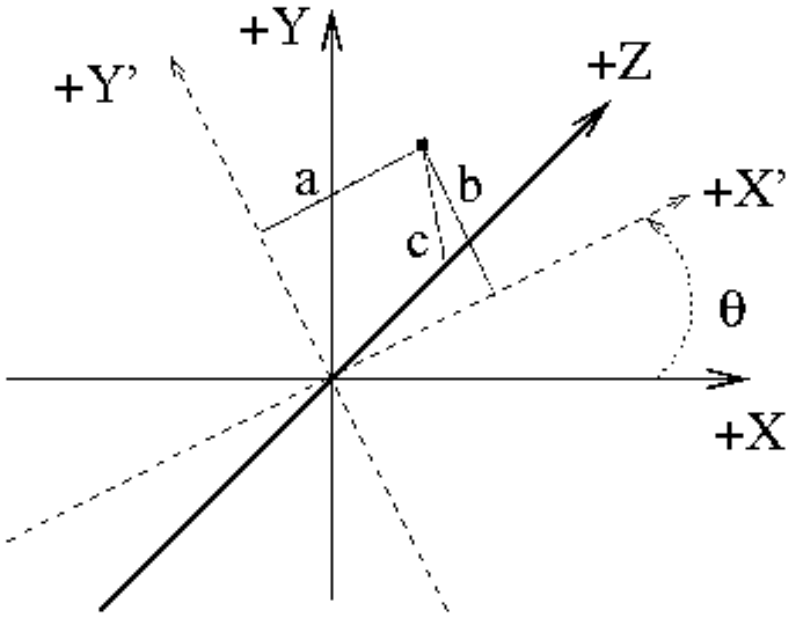
Move $\vec{p}$ to $\vec{q}$:

$$\vec{q} = \vec{p} + \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix}$$



Rotations

Rotate about *Z* axis



A lot of conventions

Here: θ positive is anti-clockwise when looking along +*Z*

- $\vec{p}$ in local (rotated) coordinates is $(a,b,c)^{T}$'
- $\vec{p}$ in global (unrotated) coordinates is
  $(a\cos(\theta)-b\sin(\theta),\ a\sin(\theta)+b\cos(\theta),c)^{T}$

Rotation Matrix Representation I

$$\vec{p} = (a,b,c)^{\mathsf{T}} \longrightarrow (a \times \cos(\theta) - b \times \sin(\theta), a \times \sin(\theta) + b \times \cos(\theta), c)^{\mathsf{T}}$$

$$R_z(\theta_z)\vec{p} = \begin{bmatrix} \cos(\theta_z) & -\sin(\theta_z) & 0 \\ \sin(\theta_z) & \cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

Much more compact and clearer!

Rotation Matrix Representation II

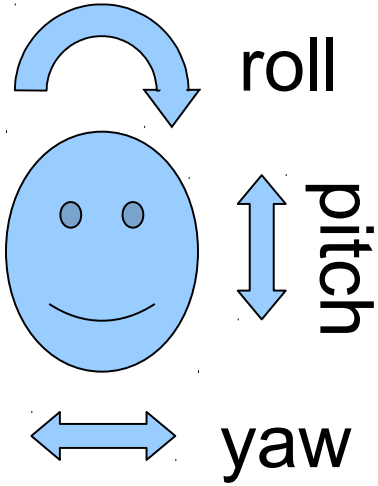
$$R_x(\theta_x)=\begin{bmatrix}1 & 0 & 0\\ 0 & \cos(\theta_x) & -\sin(\theta_x)\\ 0 & \sin(\theta_x) & \cos(\theta_x)\end{bmatrix}$$

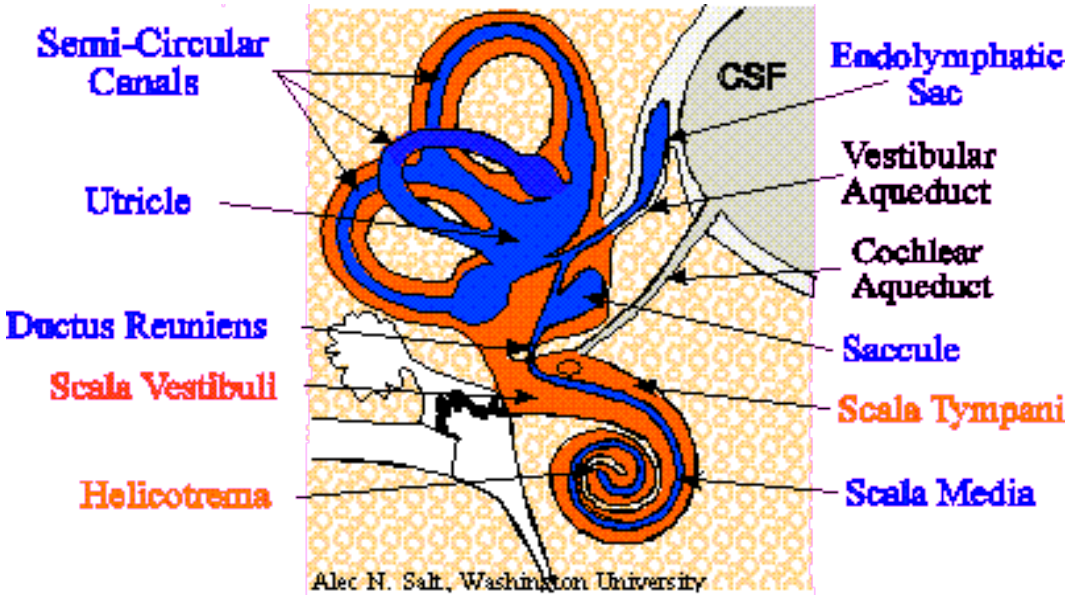
$$R_y(\theta_y)=\begin{bmatrix}\cos(\theta_y) & 0 & \sin(\theta_y)\\ 0 & 1 & 0\\ -\sin(\theta_y) & 0 & \cos(\theta_y)\end{bmatrix}$$

Other Rotation Representations

All equivalent but different parameters:

- Yaw, pitch, roll
- Azimuth, elevation, twist
  (slant, tilt, twist)
- Axis + angle
  → Quaternions
- N.B.: in mammals:





Full Rotation Specification

- Need 3 angles for arbitrary 3D rotation
- Lock & key example (axis and direction of key bit)
- Rotation angles (α, β, γ):

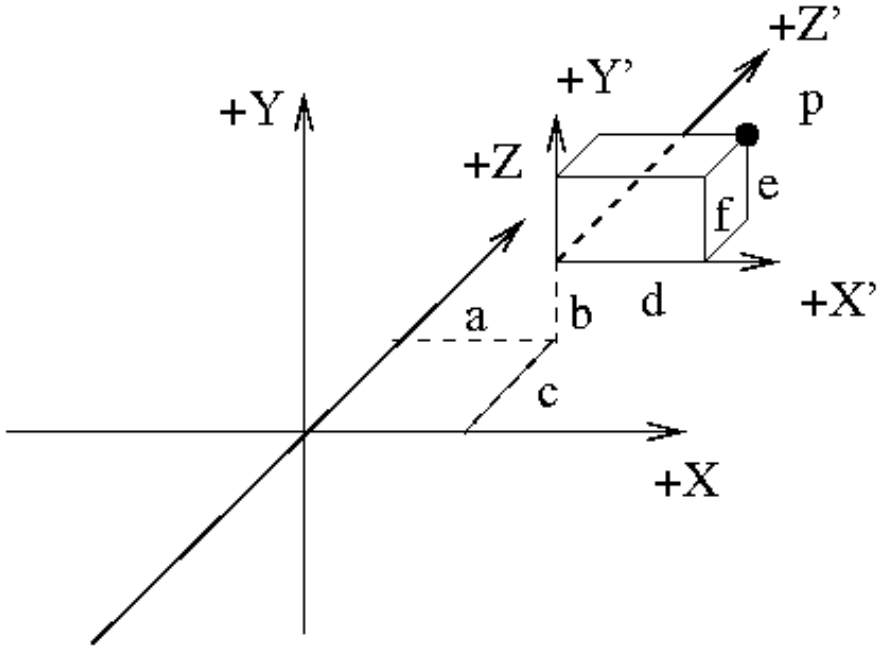
$$R(\alpha, \beta, \gamma)\, p = R_z(\gamma) R_y(\beta) R_x(\alpha)\, p$$

- Warning: rotation order by convention but must be used consistently:

$$R_z(\gamma) R_y(\beta) R_x(\alpha) \neq R_x(\alpha) R_y(\beta) R_z(\gamma)$$

Full Transformation Specification

Each connection has a new local coordinate system

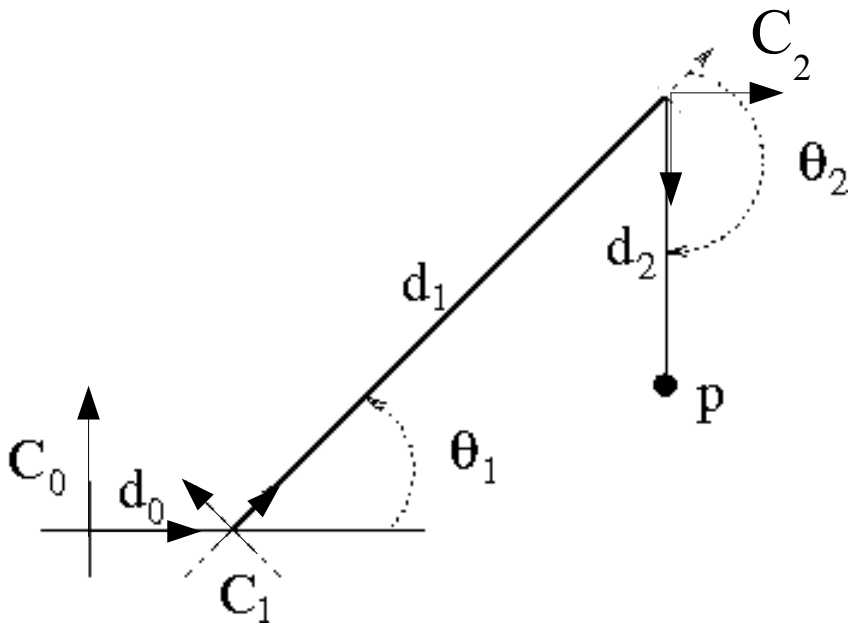


Need to specify
6 degrees
of freedom =
3 rotation + 3 translation

$$\text{transform}(\theta_x, \theta_y, \theta_z, t_x, t_y, t_z) = \text{trf}(\theta, t)$$

Kinematic Chains in 2D

p is:



in C_2: $\begin{pmatrix} d_2 \\ 0 \end{pmatrix}$

in C_1: $R(\theta_2)\begin{pmatrix} d_2 \\ 0 \end{pmatrix}+\begin{pmatrix} d_1 \\ 0 \end{pmatrix}$

in C_0: $R(\theta_1)[\,R(\theta_2)\begin{pmatrix} d_2 \\ 0 \end{pmatrix}+\begin{pmatrix} d_1 \\ 0 \end{pmatrix}]+\begin{pmatrix} d_0 \\ 0 \end{pmatrix}$

Homogeneous Coordinates I

Messy when more than 2 links, as in robot
So: pack rotation and translation into

Homogeneous coordinate matrix

Extend points with a 1 from 3-vector to 4-vector
Extend vectors with a 0 from 3-vector to 4-vector
Pack rotation and translation into 4x4 matrix:

$$H_1 = \begin{bmatrix} R(\vec{\theta}_1) & \vec{t}_1 \\ \vec{0} & 1 \end{bmatrix}$$

3 rotation parameters: $\vec{\theta}_1$
3 translation parameters: $\vec{t}_1$

Homogeneous matrices

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} \cos\beta & 0 & \sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} \cos\gamma & -\sin\gamma & 0 & 0 \\ \sin\gamma & \cos\gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

rotations

$$\begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

translation

scaling

$$\begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

projection

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

perspective

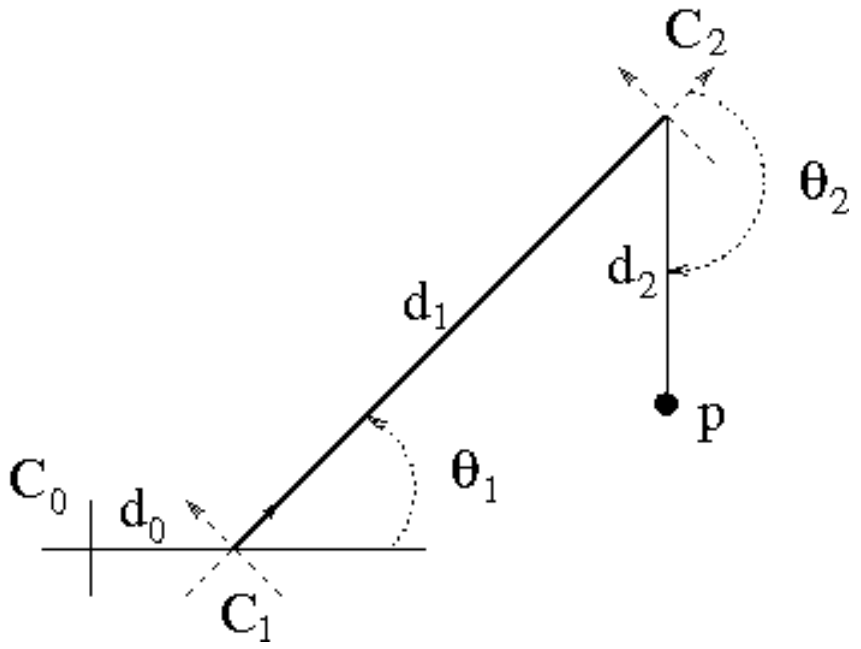
$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{d} & 0 \end{pmatrix}$$

view plane z=0, center of projection at (0,0,-d)

Isometries also represented as 6D vector: trf(α,β,γ,l_x,l_y,l_z)

Homogeneous coordinates II

$$\vec{p}^{\,*} = \begin{pmatrix} \vec{p} \\ 1 \end{pmatrix}$$



In C_2: p^*

In C_1: $H_2 p^*$

In C_0: $H_1 H_2 p^*$

Longer chains for robot arms (e.g. 6 links):

$$H_1 H_2 H_3 H_4 H_5 H_6 p^*$$

Joint geometry

Linear (prismatic) joint:
slides
parametrize one
translation direction
per joint

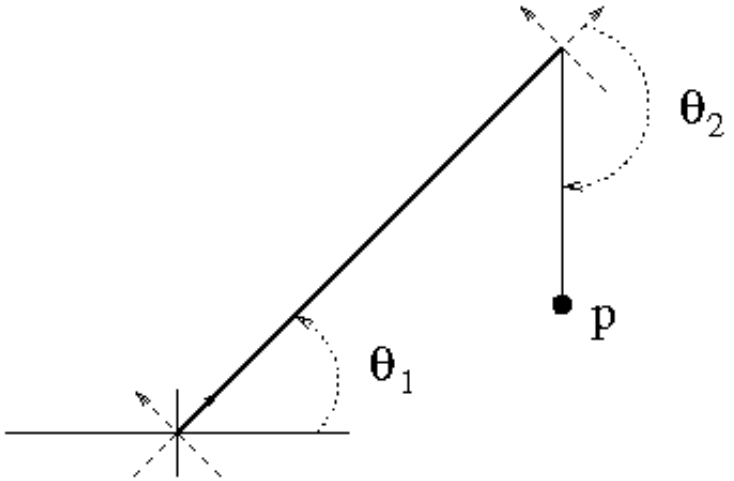
e.g. Sliding in the *x* direction

trf(0,0,0,λ,0,0)

Hinge (revolute) joint:
rotates
parametrize one
rotation angle per joint

e.g. Rotation about *x*-axis

trf(θ,0,0,0,0,0)



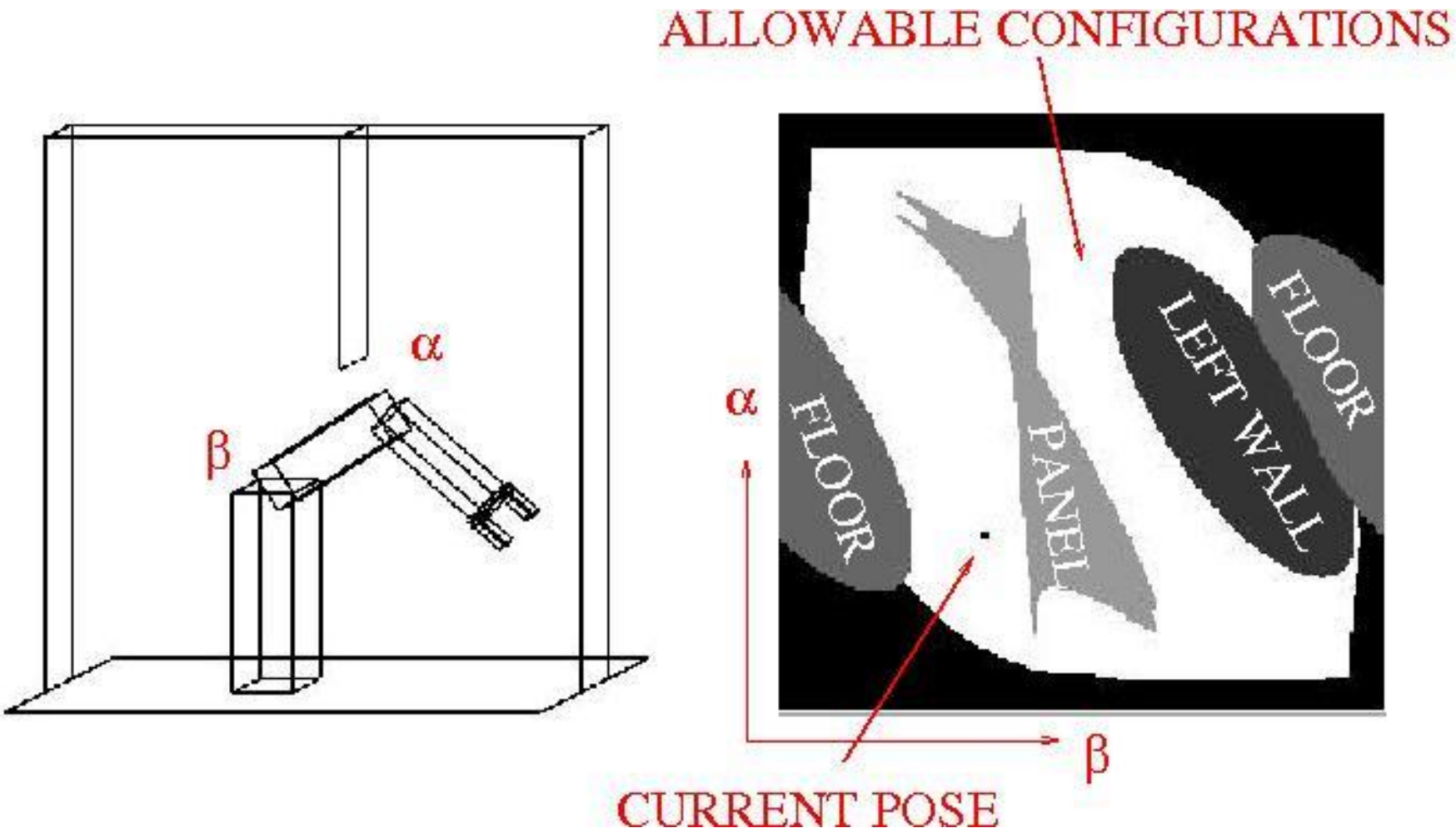
Configuration Space I

Alternative representation to scene coordinates

Number of joints = *J*
J-dimensional space
Binary encoding: *0* for invalid pose, *1* for free space
Real-valued encoding:
"distance" from goal configuration

Point in C.S. = configuration in real space
Curve in C.S. = motion path in real space

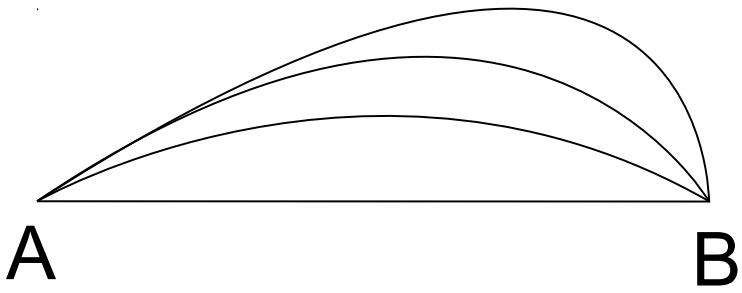
Configuration Space II



Dynamics: Trajectory Planning

What is the shortest path in configuration space?
At what speed the path is traversed?

- Cost to go (energy or wear)
- Time to goal
- Least perturbations (predictability)
- Maximal smoothness
- Minimal intervention



Forward and Inverse Kinematics

Forward:

Given joint angles, find gripper position
Easy for sequential joints in robot arm:
just multiply matrices

Inverse:

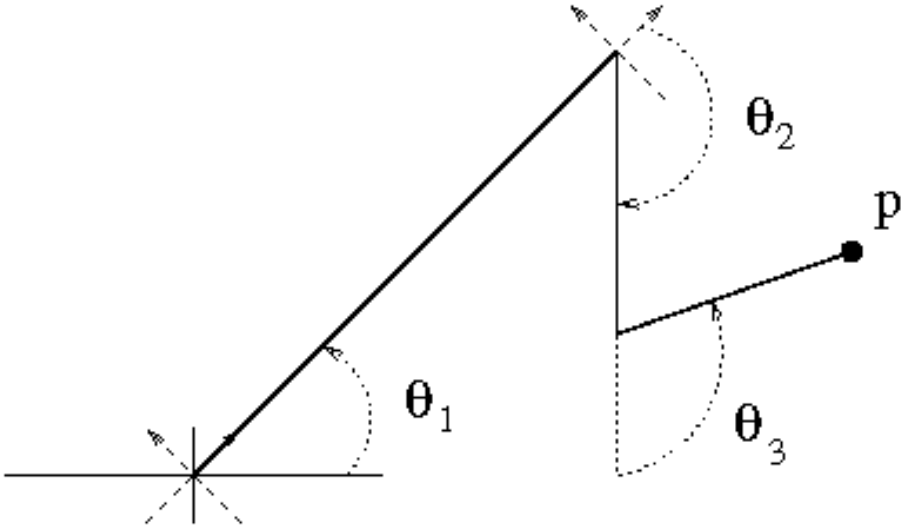
Given desired gripper position, find joint angles
Hard for sequential joints – geometric reasoning

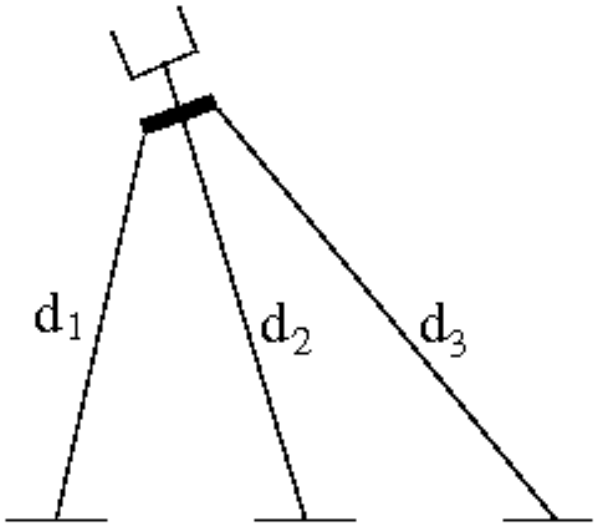
Sequential & Parallel Mechanisms

Simplified into 2D

Serial manipulator vary: $\theta_1, \theta_2, \theta_3$

Parallel manipulator vary: d_1, d_2, d_3



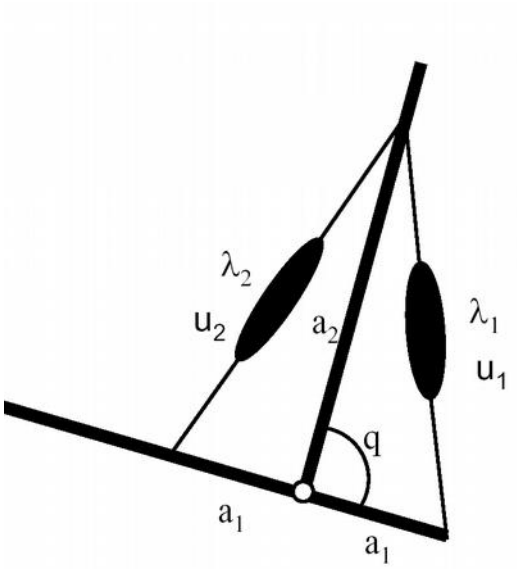


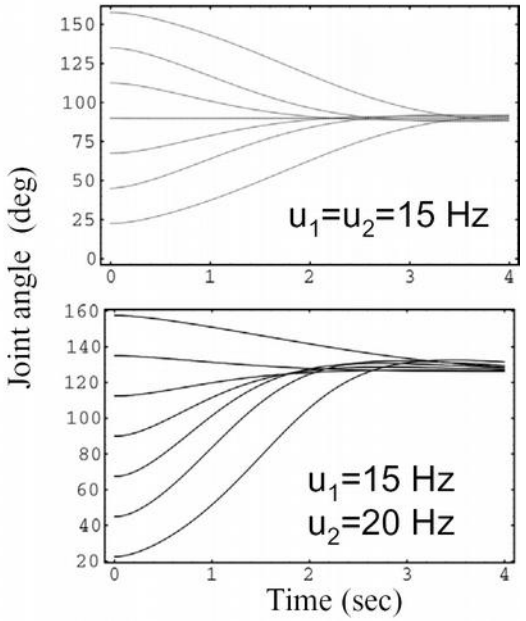
Computing Positions & Parameters

	Serial	Parallel
Forward (param->position)	Easy (just multiply matrices)	Hard (messy robot specific geometry)
Inverse (position->param)	Hard (messy robot specific geometry)	Easy (just read off lengths from gripper position)

N.B.: Biological motor control

Equilibrium point hypothesis





A. G. Feldman 66
Bizzi et al. 78
Gribble et al. 98

- Extensors and flexors controlled by different pathways:
- Force-field experiments: violation of equifinality at variations of velocity-dependent load (Hinder & Milner 03)

→ Solution: Internal dynamics models (Shadmehr & Mussa-Ivaldi 94)

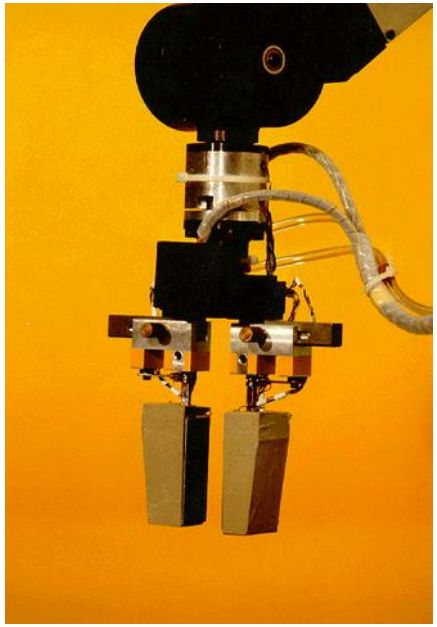
Specifying Robot Positions

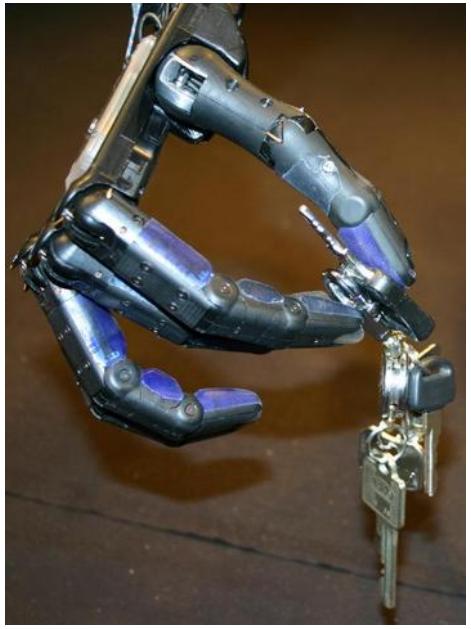
1. Actuator level: specify voltages that generate required joint angles.
2. Joint level: specify joint angles and let system calculate voltages.
3. Effector level: specify tool position and let system compute joint angles.
4. Task level: specify the required task and let the system compute the sequence of tool positions

Most robot programming is at levels 2 and 3.

Grippers and Grasping

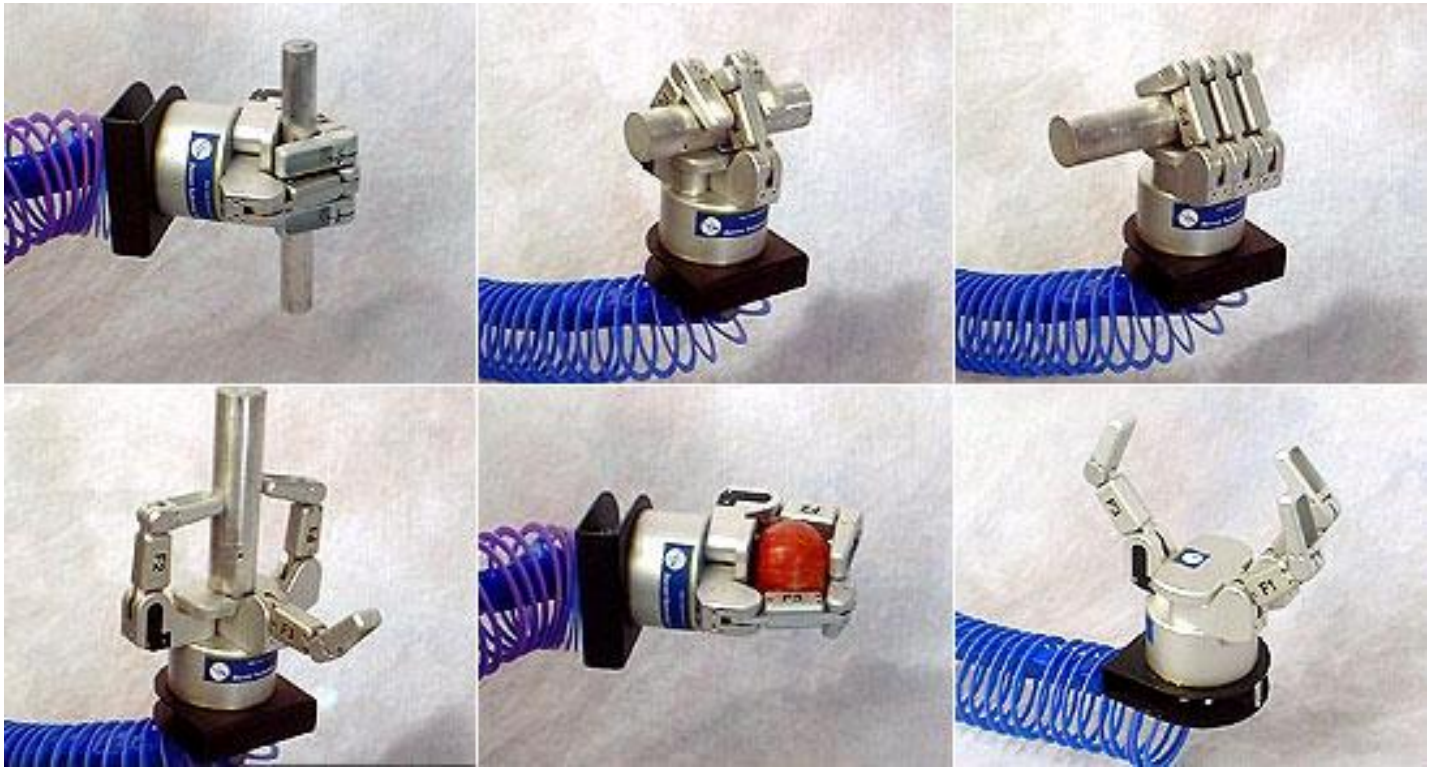
- Gripper: special tool for general part manipulation
- Fingers/gripper: 2, 4, 5
- Joints/finger: 1, 2, 3





Your hand: 4 fingers * 4 DoF + thumb * 5 DoF+ wrist * 6 DoF = 27 DoF (22 controllable DoF).

Barrett Technology Hand

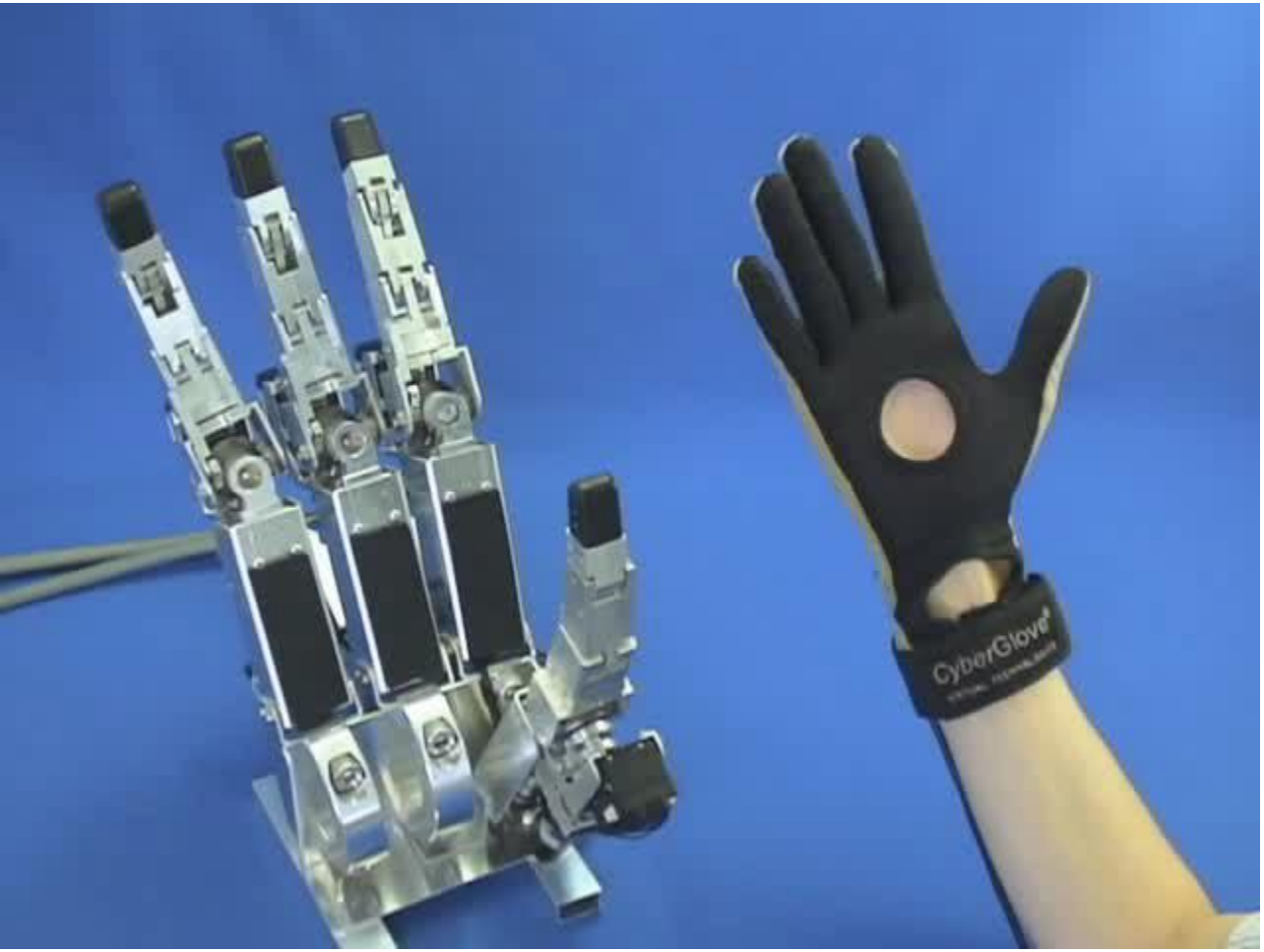


2 parallel fingers (spread uniformly)

1 opposable finger

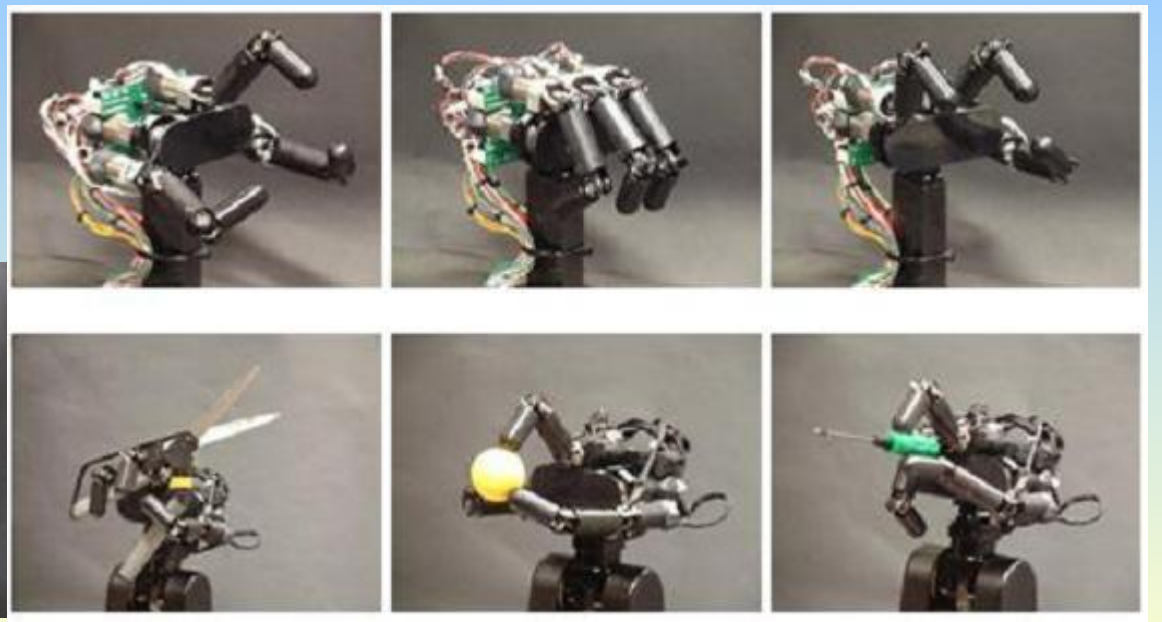
DoF: 4 fingers (2 finger joints bend uniformly)

Shadow Dextrous Robot Hand



http://www.shadowrobot.com/hand/

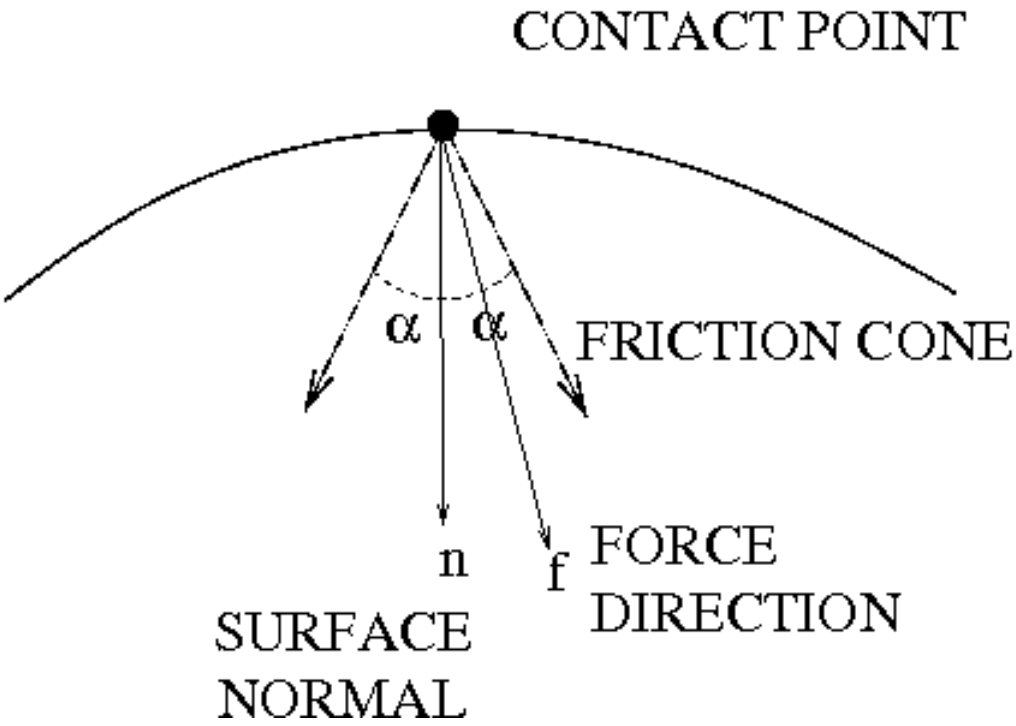
High-Speed Robotic Hand



Ishikawa Komuro

Finger Contact Geometry

- Coefficient of friction at fingertip: $\mu \in [0,1]$
- Surface normal: Direction perpendicular to surface: $\vec{n}$
- Friction cone: Angles within $\alpha = \cos^{-1}(\mu)$ about surface normal
- Force direction: f direction in which finger pushed
- No-slip condition: $\vec{f} \cdot \vec{n} \geq \mu$



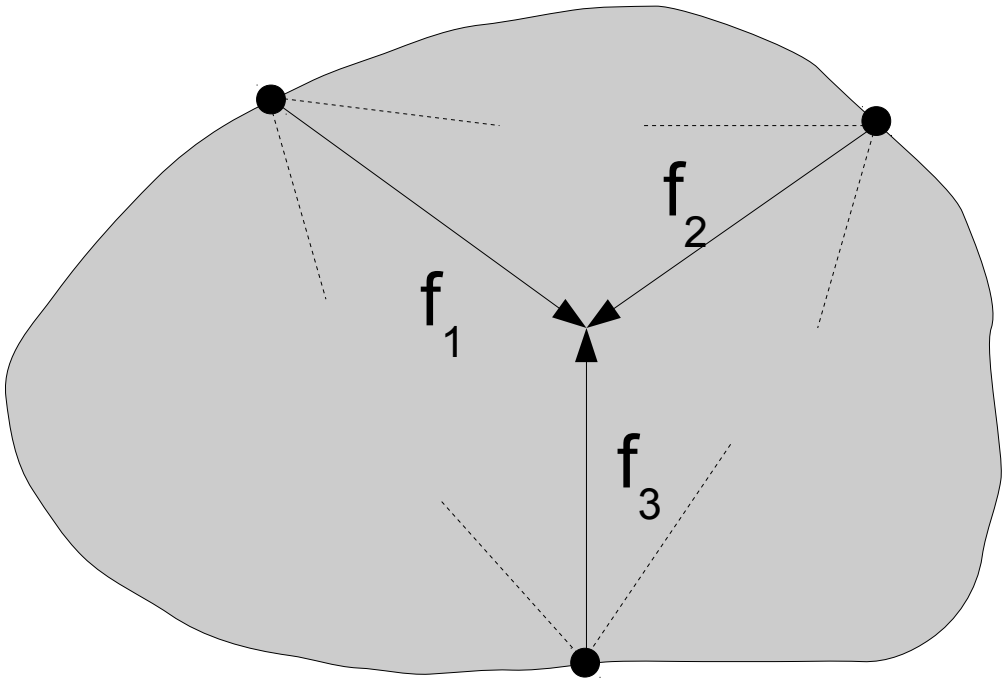
Force Closure

Need balanced forces or else object twists

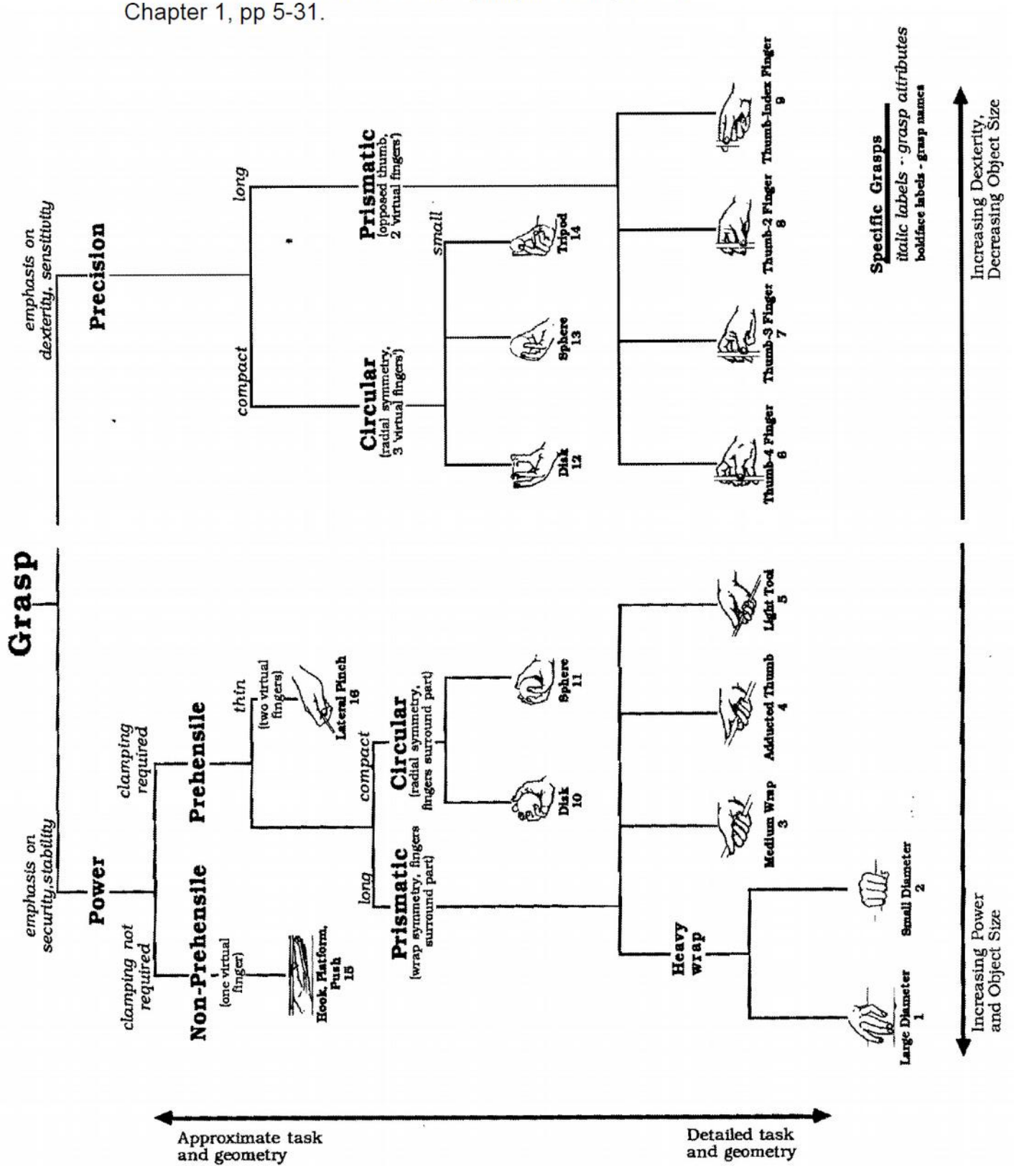
2 fingers – forces oppose: $\vec{f}_1 + \vec{f}_2 = 0$

3 fingers – forces meet at point: $\vec{f}_1 + \vec{f}_2 + \vec{f}_3 = 0$

Force closure: point where forces meet lies within 3 friction cones otherwise object slips



MR Cutkosky and RD Howe, "Human Grasp Choice and Robotic Grasp Analysis," in ST Venkataraman, T Iberall, editors, *Dextrous Robot Hands,* Springer-Verlag,1990, Chapter 1, pp 5-31.



Other Grasping Criteria

Some heuristics for a good grasp:

- Contact points form nearly equilateral triangle
- Contact points make a big triangle
- Force focus point near Center of Mass (CoM)

Grasp Algorithm

1. Isolate boundary
2. Locate large enough smooth graspable sections
3. Compute surface normals
4. Pick triples of grasp points
5. Evaluate for closure & select by heuristics
6. Evaluate for reachability and collisions
7. Compute force directions and amount
8. Plan approach and finger closing strategy
9. Contact surface & apply grasping force
10. Lift (& hope)

Kinematics Summary

1. Need vector & matrix form for robot geometry
2. Geometry of joints & joint parameters
3. Forward & inverse kinematics
4. Degrees of freedom
5. Grippers & grasping conditions

IVR: Sensing Self-Motion

Robotics Lecture 9

informatics

Overview

1. Proprioception
2. Sensors for self-sensing in biological systems
   - proprioception
   - vestibular system
3. Sensors for self-sensing in robotic systems
   - velocity and acceleration sensing
   - force sensing
   - position sensing
4. Vision as proprioception (next time)

1. Proprioception: Why robots need self-sensing

- For a robot to act successfully in the real world it needs to be able to perceive the world and its relation to the world.
  - The state of the robot is not entirely up to the robot itself, but also reflects external events. Thus, information about the "body" is an important source of information about the world.
  - Another use of proprioceptive information is stabilisation and smoothing of planned movements against perturbations.
- In particular, to control its own actions, the robot needs information about the position and movement of its body and parts
- Our body contains at least as many sensors for our own movement as it does for signals from the world.

Odometry: Using self-sensing to estimate position

- Odometry is an internal mechanism that estimates the robot's position relative to a starting point by adding up the distance travelled (from gr. *hodos*, meaning "travel", "journey")
- Know present position (x, y, ϕ)
- Know how much wheels rotate or count steps
- New position = old position + change of position based on commanded motion
- Computationally simple
- Dead-reckoning is similarly based on the measurement of speed perhaps including a heading sensor (compass)

Odometry: Discussion

- Less reliable when done based on motor commands: Motors may be inaccurate and low-level controllers may compromise the high-level control command
- Use proprioception (shaft encoders) instead of an "efference copy"
- Calibration reduces systematic errors
- Prone to "random" errors: wheel slip, surface roughness, wall contacts, noise, imprecise measurement; errors tend to accumulate
- Remaining in sync with the environment: Some feature tracking needed. Combine with landmark/beacon-based navigation
- The angular component of the position is particularly critical: a compass may help
- Movement model tends to be less precise for curved trajectories: Constrain behaviour to a few types ("straight" and "turn in place") to make calibration easier.

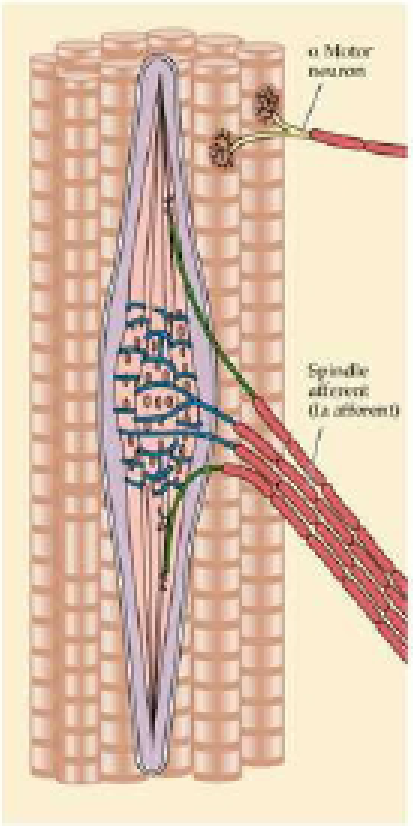
Sensors for self-sensing in biological systems

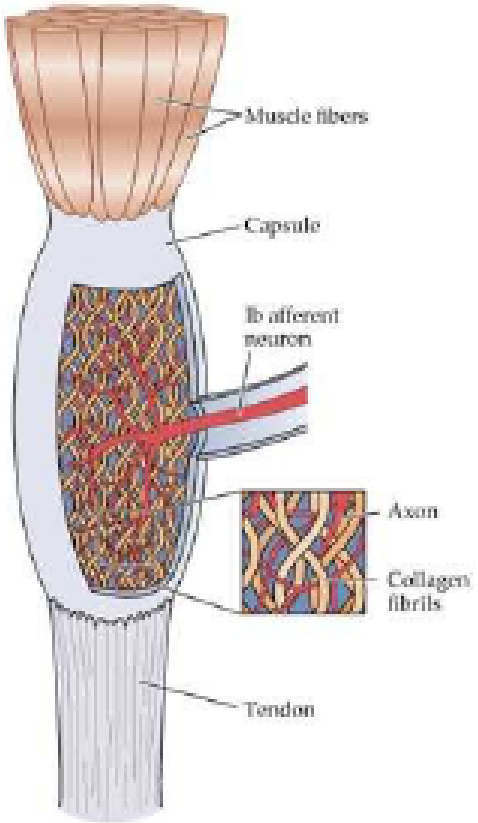
Interoception in humans

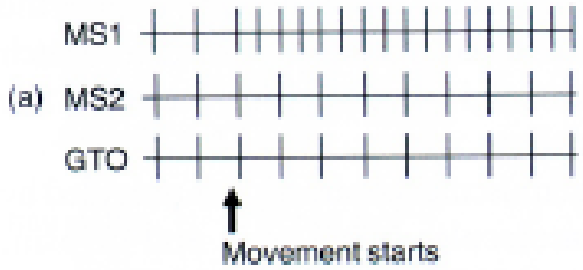
- Mechanoreceptors: Occur often in two types: deviations to below and above standard
  - in the lungs sense stretch and contribute to the control of the respiration rate
  - in brain arteries are involved in blood pressure regulation (baroreceptors)
  - in the gastrointestinal tract sense gas distension
  - in the bladder and rectum report fullness
  - various types of exteroceptive mechanoreceptors
- Nociception (sensing pain; different from but intertwined with sensing of pressure and touch)
- Thermoception in the brain (hypothalamus) regulating body temperature
- Chemoreceptors measure carbon dioxide and oxygen levels in the brain, presences of hormons, salt, sugar etc. in the blood
- Chronoception?

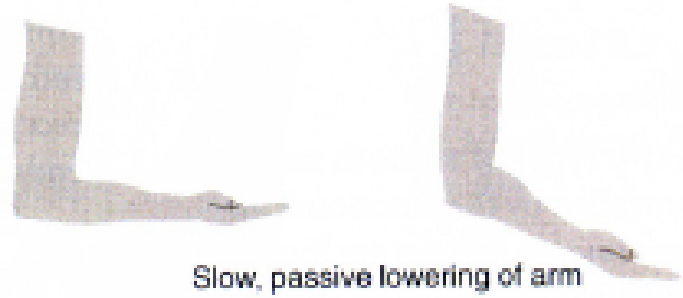
Proprioception: Detecting our own movements

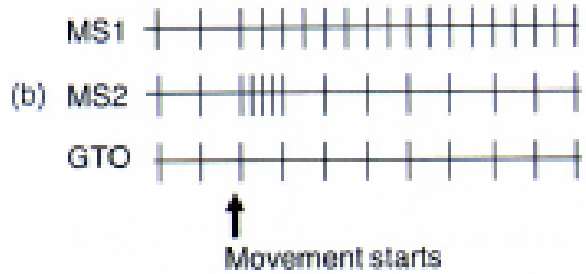
- To control our limbs we need feedback: Kinesthesia
- Muscle spindles
  - where: length
  - how fast: rate of stretch
- Golgi tendon organ
  - how hard: force
- Pressure sensors in skin (e.g. finger movements)
- Pinocchio illusion

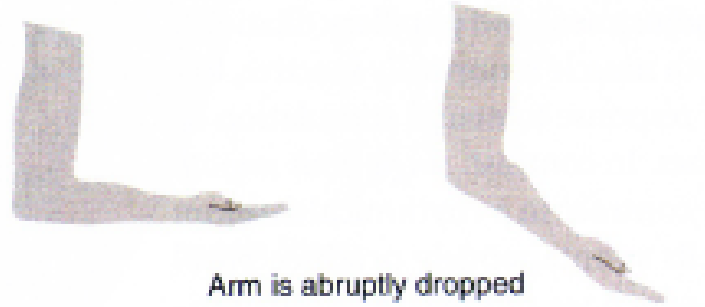


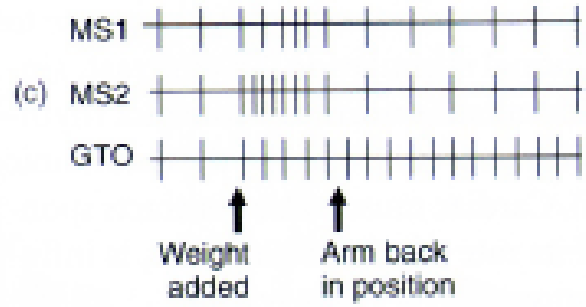


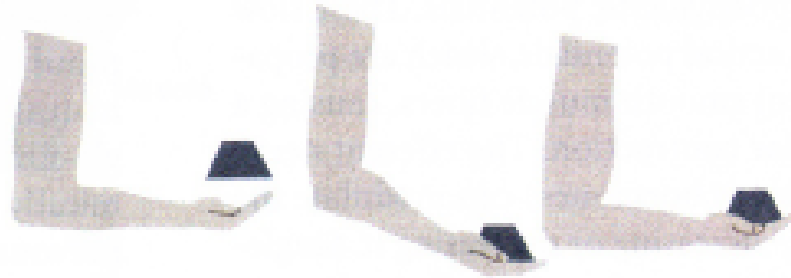

MS1
(a) MS2
GTO
Movement starts


Slow, passive lowering of arm


MS1
(b) MS2
GTO
Movement starts

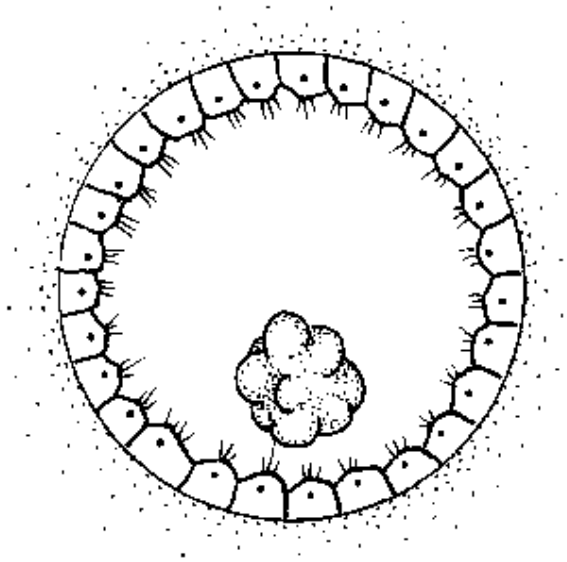

Arm is abruptly dropped


MS1
(c) MS2
GTO
Weight added
Arm back in position



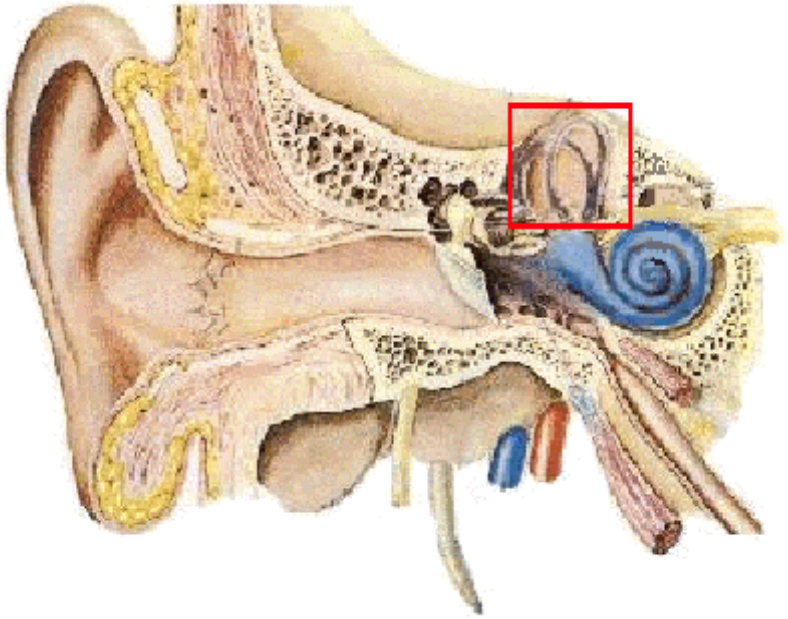
The vestibular system

- Detection of whole-body motion: Vestibular system based on statocysts (or otolithic organ in vertebrates)
- Statoliths (otoliths) are calcium nodules affected by gravity (or inertia during motion) cause deflection of hair cells that activate neurons



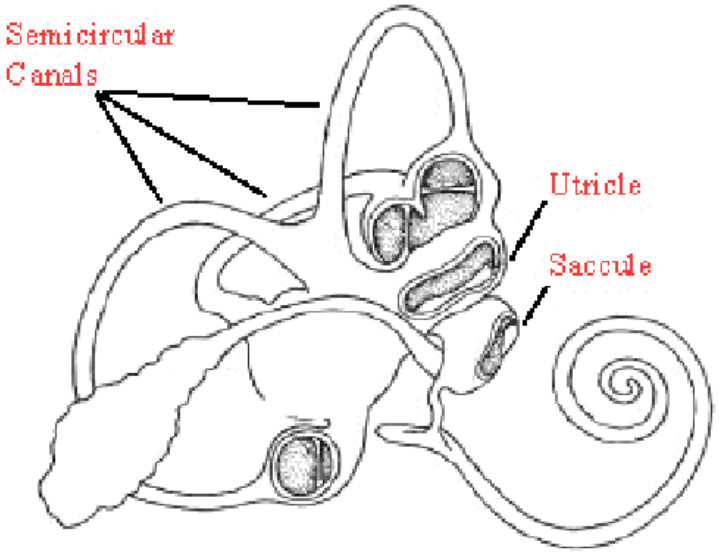
Note that sometimes the term *proprioception* is reserved for muscle and joint sensors. The integration of this and the vestibular inputs is then called *kinesthetic sense*.

Self-sensing in animals



Vestibular System

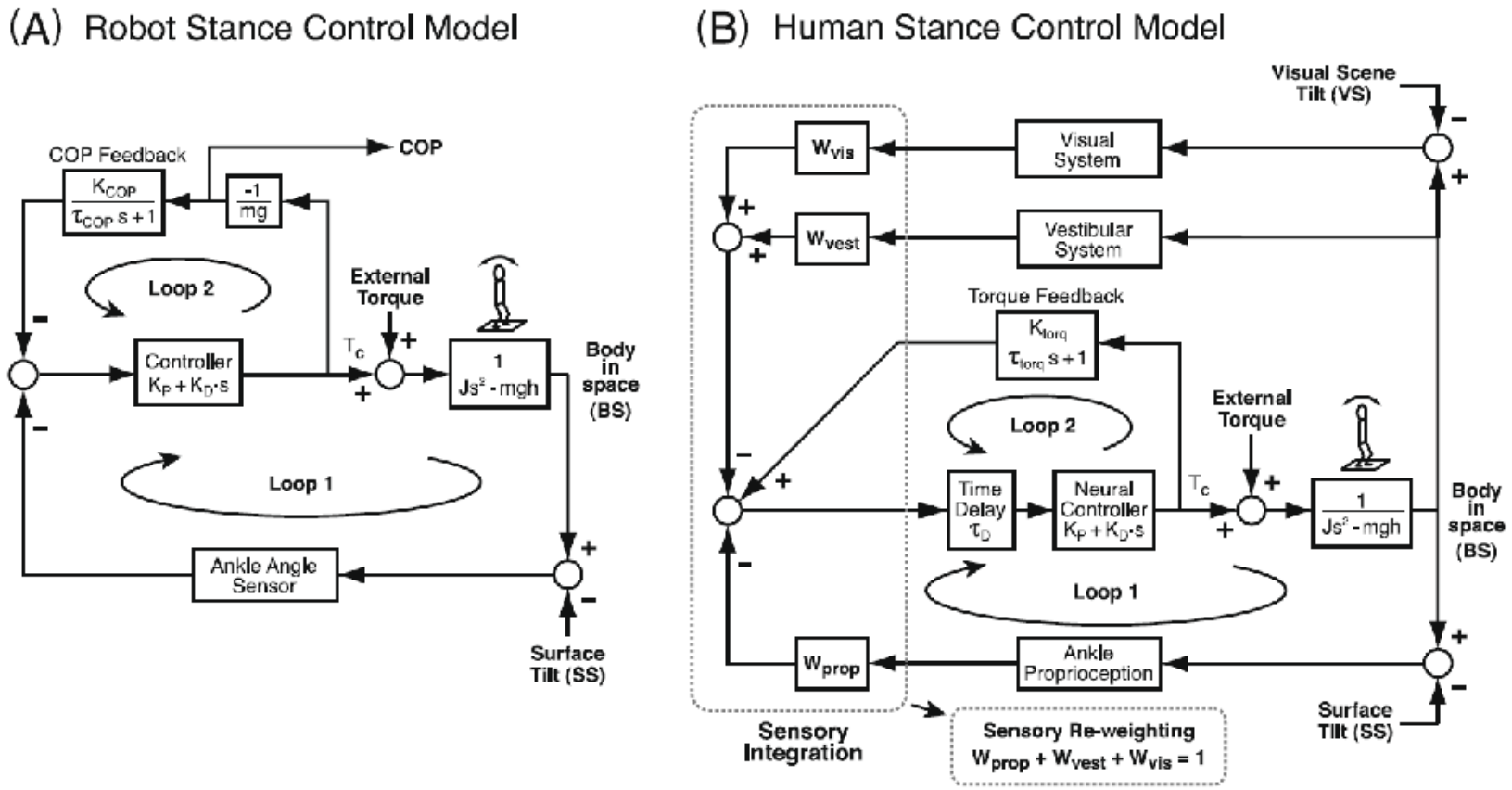
- Linear acceleration detected by Utricle (horizontal) and Saccule (vertical) based on otoliths
- Dysfunction causes vertigo



- Semicircular canals detect rotary acceleration in three orthogonal axes

Used e.g. in the fast vestibular-ocular reflex for eye stabilisation

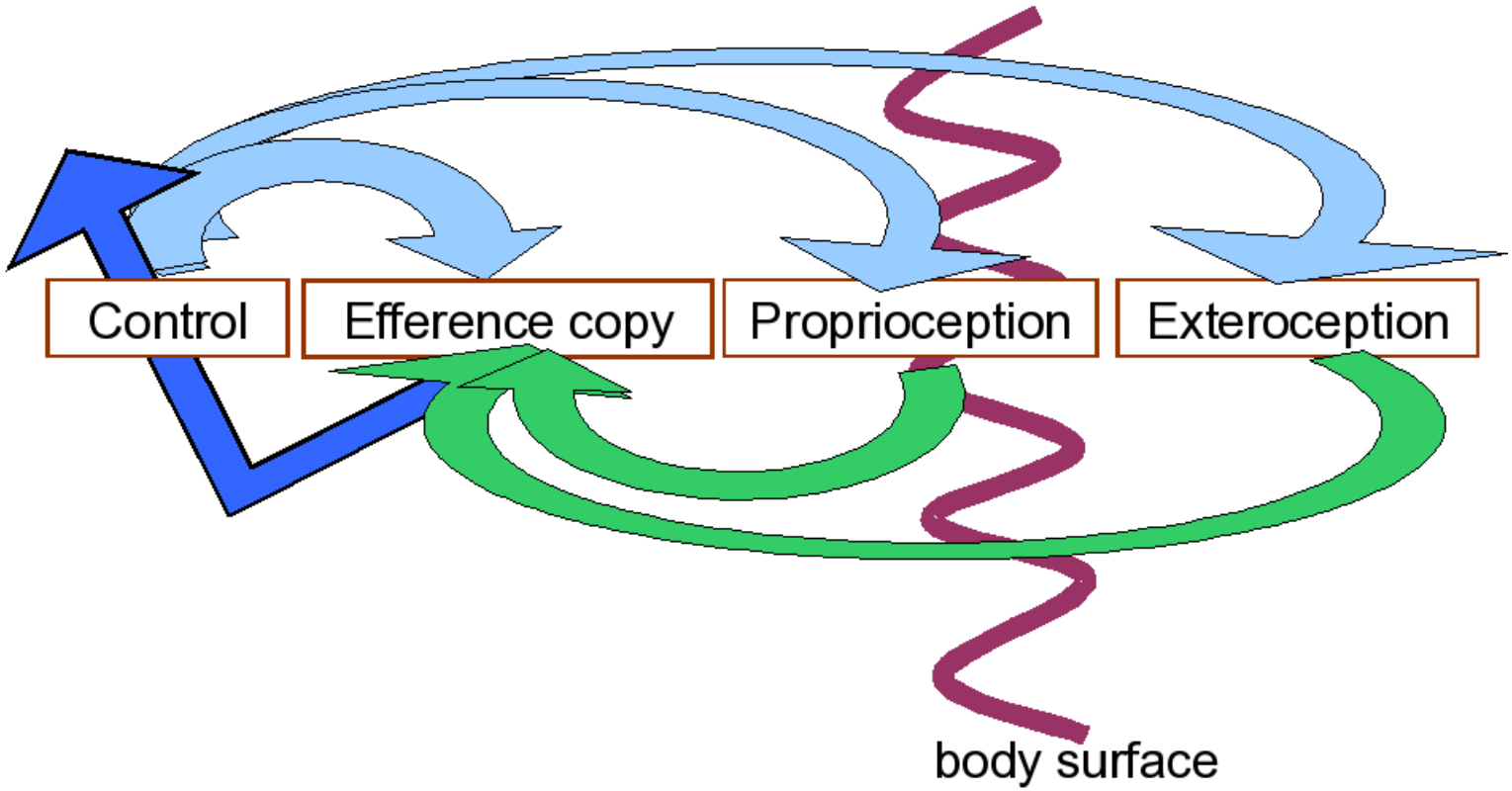
Relation to control theory



Robert J. Peterka (2009) Comparison of human and humanoid robot control of upright stance. Journal of Physiology – Paris 103, 149–158

COP: center of pressure; optic flow and gyroscopes are used too in robot stance control

Using proprioceptive information: Efference copy



Efference copy

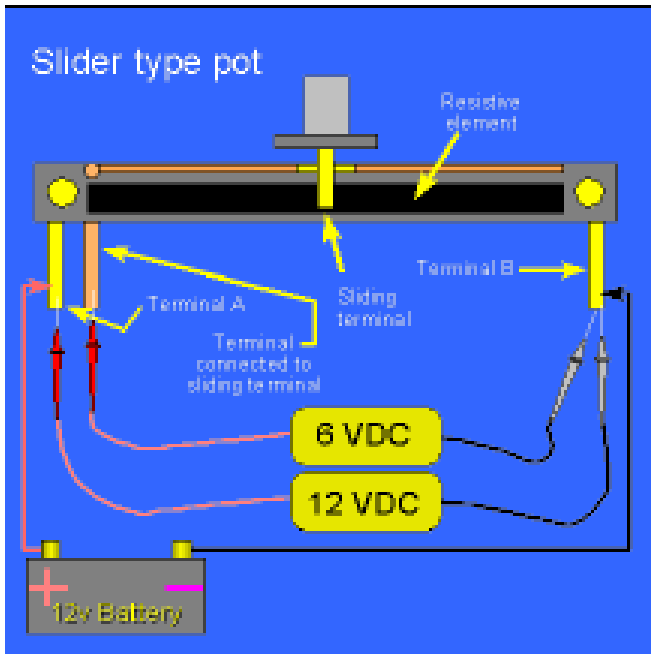
- A forward model predicting the effects of own actions
- Distinction between self and non-self (Tickling!)
- Anticipating the self-movement-induced changes of the electric sensations in weakly electric fish
- Grip force changes without time lag for known loads
- Efference copy is quite unreliable, not clearly localisable in the brain, and has low resolution:
- Gaze stabilisation
  - Visual input (highest priority)
  - self-sensing head and body movement by vestibular system
  - efference copy at active movements

Sensors for self-sensing in a robot

For a robot:

Need to sense motor/joint positions with e.g.:

- Potentiometer (current measures position)
- Optical encoder (counts axis turning)
- Servo motor (with position feedback)



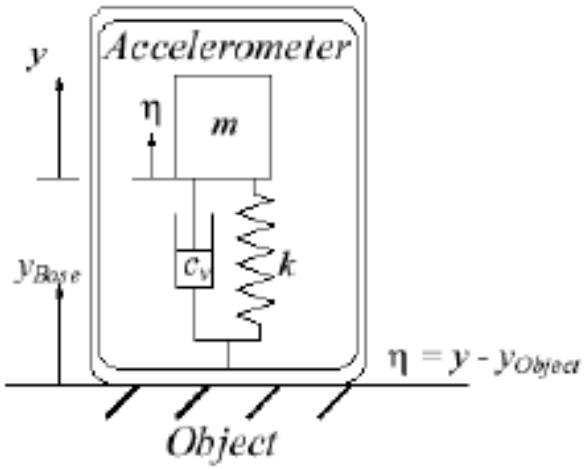
Proprioception for a robot

- Velocity by position change over time or other direct measurement: Tachometer, e.g. using principle of dc motor in reverse: voltage output proportional to rotation speed
- Acceleration: could use velocity over time, but more commonly, sense movement or force created when known mass accelerates, i.e. similar to statocyst
- Common uses of proprioceptive measurements are for battery monitoring, current sensing, and heat monitoring.
- The robot measures a signal originating from within using proprioseptive sensors. Responsible for monitoring self maintenance and controlling internal status. Often same modality as contact sensors (in contrast to distance sensors)

Kinestetics for a robot

Accelerometer:
measures displacement of weight due to inertia

Gyroscope:
uses conservation of angular momentum



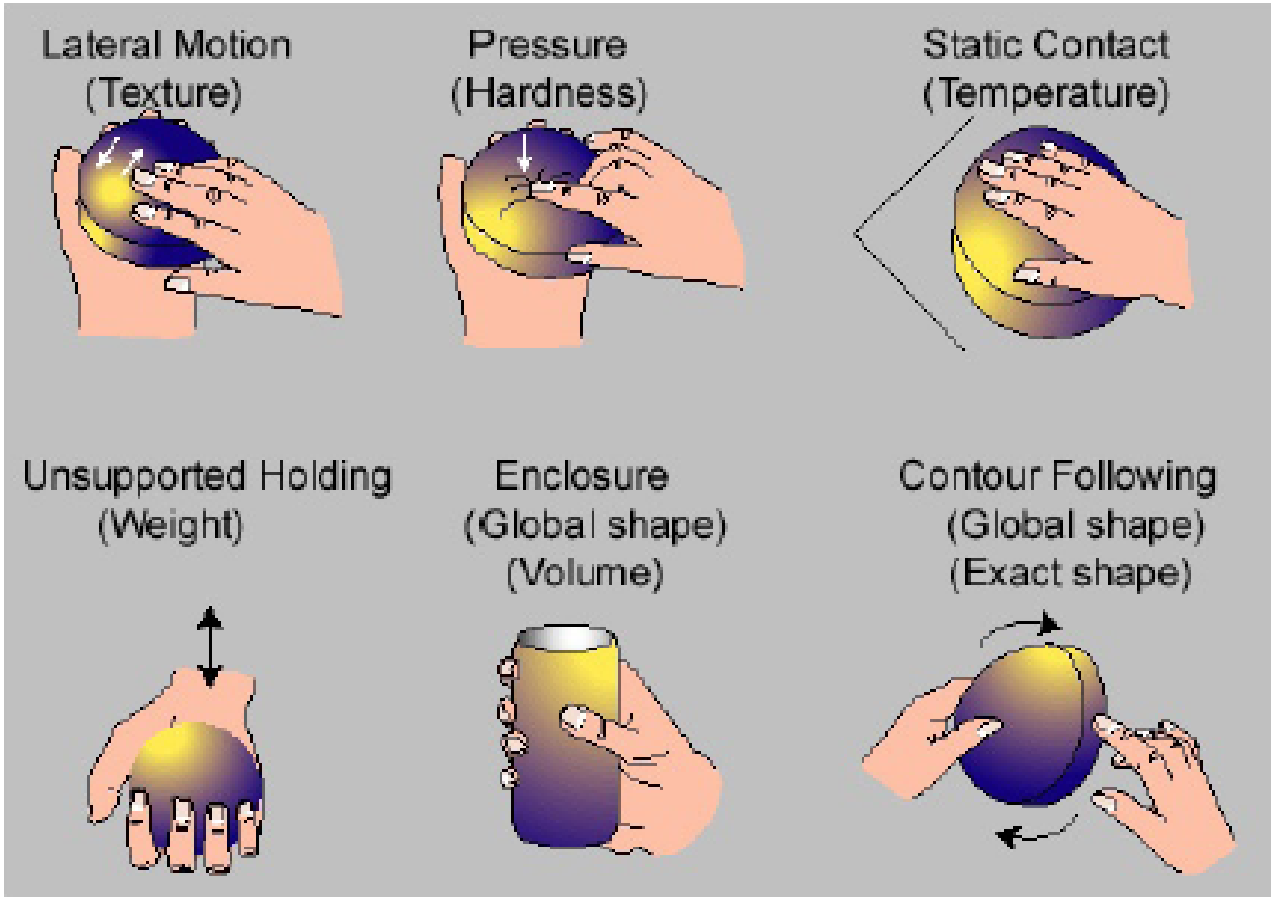
- There are many alternative forms of these devices, allowing high accuracy and miniaturisation (e.g. ceramic piezo gyros, MEMSs)

Inertial Navigation System (INS)

- Based on an Inertial Measurement Unit (IMU)
- Three accelerometers for linear axes
- Three gyroscopes for rotational axes (or to stabilise platform for accelerometers)
- By integrating over time can track exact spatial position
- Viable in real time with fast computers
- But potential for cumulative error

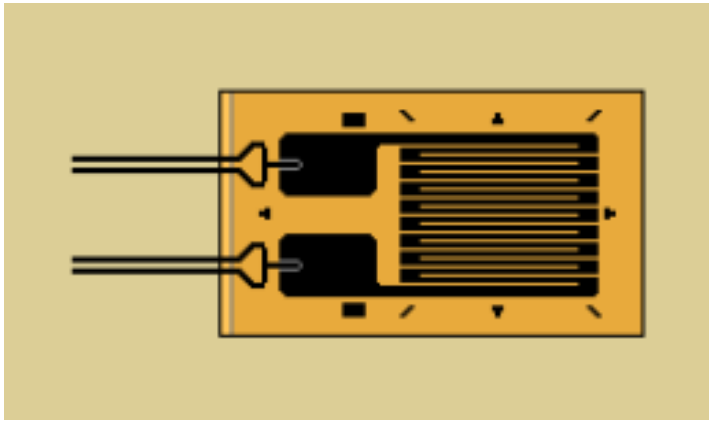
Haptics as exproprioception

Combining muscle & touch sense



Force sensing in a robot

- Spring and potentiometer
- Resistance change with deformation: Strain gauge $\rightarrow$
- Piezoelectric – charge created by deformation of quartz crystal (n.b. this is transient)
- Nanowire active matrix (Nature Materials 2010) for artificial skin
- BioTac sensor (USC, 2012) $\rightarrow$



More sensors for a robot:

Various other sensors may be used to measure the robot's position and movement, e.g.:

- Tilt sensors
- Shaft Encoders
- Compass
- Global Positioning System (GPS)
- May use external measures e.g. camera tracking of limb or robot position

Some issues for sensors

- What range, resolution and accuracy are required? How easy to calibrate?
- What speed (i.e. what delay is acceptable) and what frequency of sampling?
- How many sensors? Positioned where?
- Is information used locally or centrally?
- Does it need to be combined? How?
- Computational complexity

Using sense data

- Similar issues as with other sensors: Noise tolerance, reliability, context dependence
- Bayesian approaches:
  - testing hypotheses
  - combination of proprioception and exteroception
  - sensor fusion
- Evolutionary robotics: Adjust configuration of sensors

Proprioceptive control in BigDog

Boston Dynamics

- Proprioceptive sensors: linear pots, load cells, current sensors (4 of each per leg = 48)
- Homeostatic sensors: temperature sensors, sensors for oil flow, oil pressure, engine rpm, battery voltage
- Exteroceptive: gyro, 3 stereo vision systems, 1 LIDAR

Total: 69, most of which relate to the state of the robot

Marko B. Popovic (2013) Biomechanics and Robotics.

Summary

- Have discussed a variety of natural and artificial sensors for self motion
- Have hardly discussed how the transduced signal should be processed to use in control for a task
- Proprioceptive information needs to be continuously reconciled with exteroceptive information

IVR: Visual Servoing

R. Fisher, M. Herrmann

informatics

Overview

- Model-based control
- Visual servoing task
- What is a Jacobian?
- Visual servoing task
- Implications for robot control

PROBLEM: Guiding a robot to a target based on images

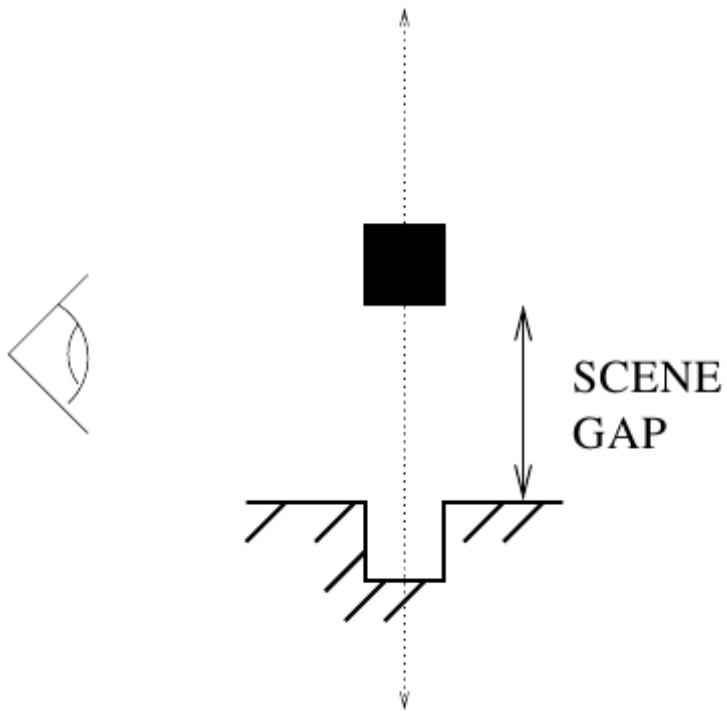
APPLICATIONS:

- Assembly robotics: Component insertion
- Mobile robotics: Docking

$\Longrightarrow$ VISUAL SERVOING THEORY

QUESTION

If only vertical robot motion possible (and sufficient) to reduce the gap:



How to get peg exactly into hole?

APPROACH 1: MODEL DRIVEN

KNOWN:

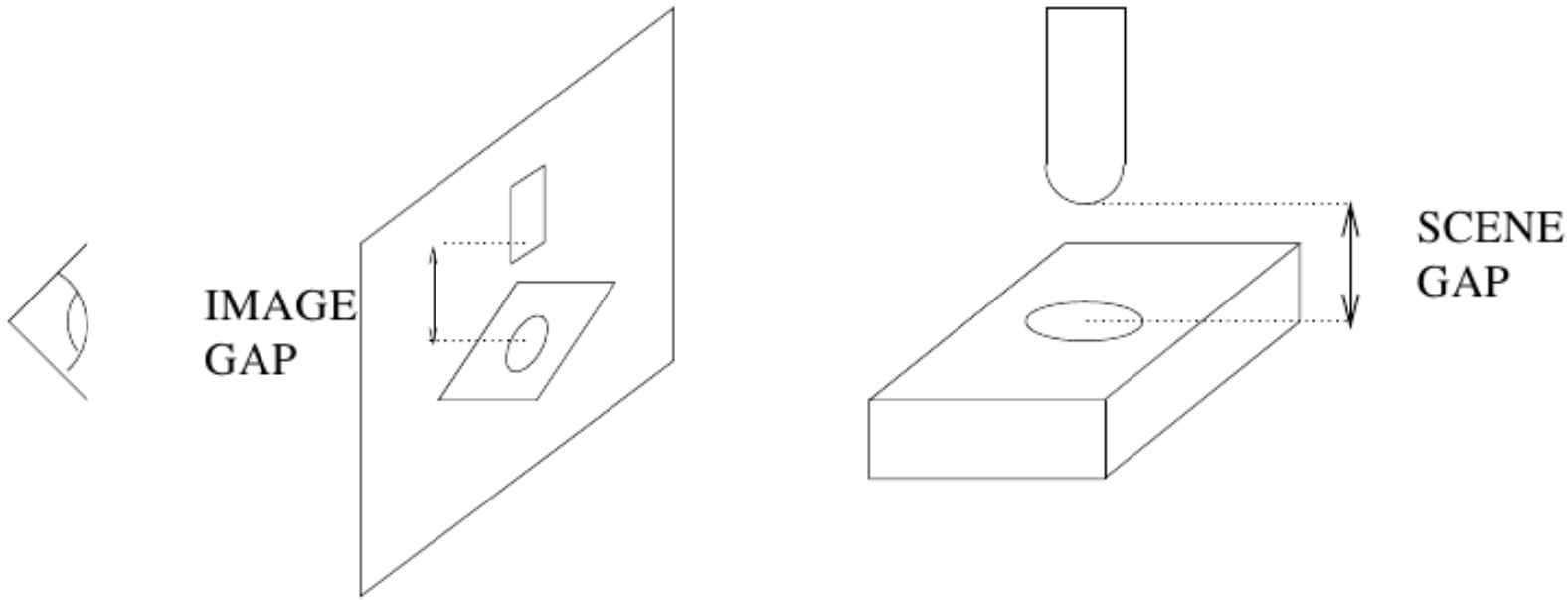
- Robot manipulation model
- Part & robot position known
- Target model
- Camera calibrated

BUT: errors exist and combine - noise, unknowns, gear backlash

SO: need very accurate calibration & mechanisms

TRY 2: VISUAL SERVOING

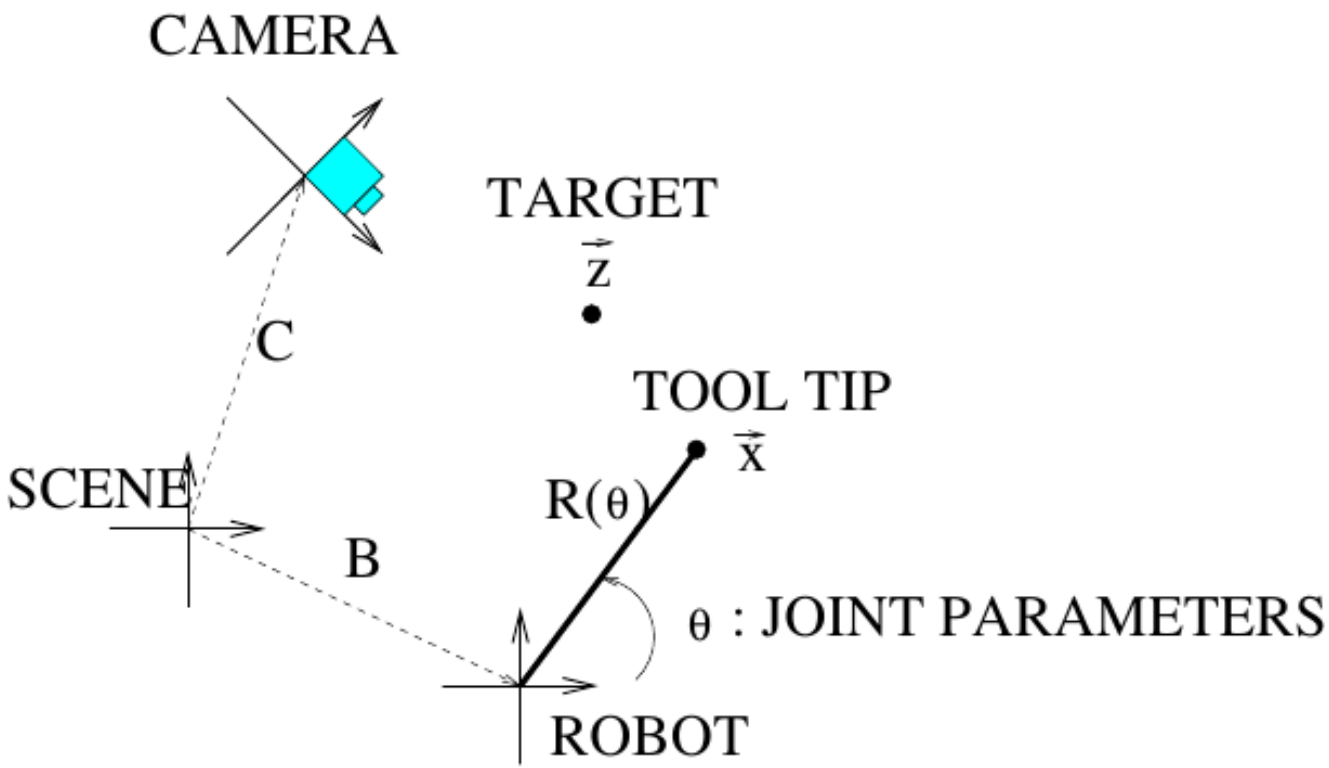
Move so that observed positions directly link to task distances



- Incrementally move to reduce image gap
- Don't know exact positions

MORE FORMAL MODEL

CONSIDER:



VARIOUS QUANTITIES

B - ROBOT POSITION WRT SCENE

C - CAMERA POSITION WRT SCENE

P - CAMERA PROJECTION MODEL

$\vec{\theta}$ - ROBOT JOINT PARAMETERS

$R\left(\vec{\theta}\right)$ - TOOL POSITION WRT ROBOT

LINKING EQUATION

IF:

- $\vec{x}$ - TOOL TIP WRT TOOL
- $\vec{z}$ - TARGET WRT SCENE
- $\vec{a}$ - IMAGE OF TOOL TIP
- $\vec{b}$ - IMAGE OF TARGET

THEN (as a concatenation of appropriate transformations):

- $\vec{a} = PC^{-1}BR(\theta)\,\vec{x}$
- $\vec{b} = PC^{-1}\vec{z}$

APPROACH 1: MODEL DRIVEN

Calibrate P, C, B, $R\left(\vec{\theta}\right)$ accurately

SOLVE: $\vec{\theta} = \vec{f}(\vec{z}, \vec{x}, P, C, B)$

Hard analytics

Not always solvable

APPROACH 2: VISUAL SERVOING

How much to move joints ($\Delta\vec{\theta}$) to reduce target error $\vec{\Delta} = \vec{a} - \vec{b}$?

Visually estimate $\vec{f}(\cdots)$ such that $\Delta\vec{\theta} \doteq \vec{f}(\vec{\Delta})$

Use $\Delta\vec{\theta}$ to partially approach target and recompute $\vec{\theta}$

Iterate

ESTIMATING $\vec{f}()$

Move robot joint i slightly from θ_i to $\theta_i + \varepsilon$

Observe tool tip moves slightly from $\vec{a}$ to $\vec{a} + \vec{\delta}_i$

Compute:

$$\frac{\partial \vec{a}}{\partial \theta_i} \doteq \frac{\left(\vec{a} + \vec{\delta}_i\right) - \vec{a}}{(\theta_i + \varepsilon) - \theta_i} = \frac{\vec{\delta}_i}{\varepsilon}$$

Repeat for all i to estimate JACOBIAN matrix

$$J = \frac{1}{\varepsilon}\left[\vec{\delta}_1, \ldots, \vec{\delta}_N\right]$$

for N joints.

USING THE JACOBIAN

Linear theory:

$$\vec{\Delta} \doteq J \Delta \vec{\theta}$$

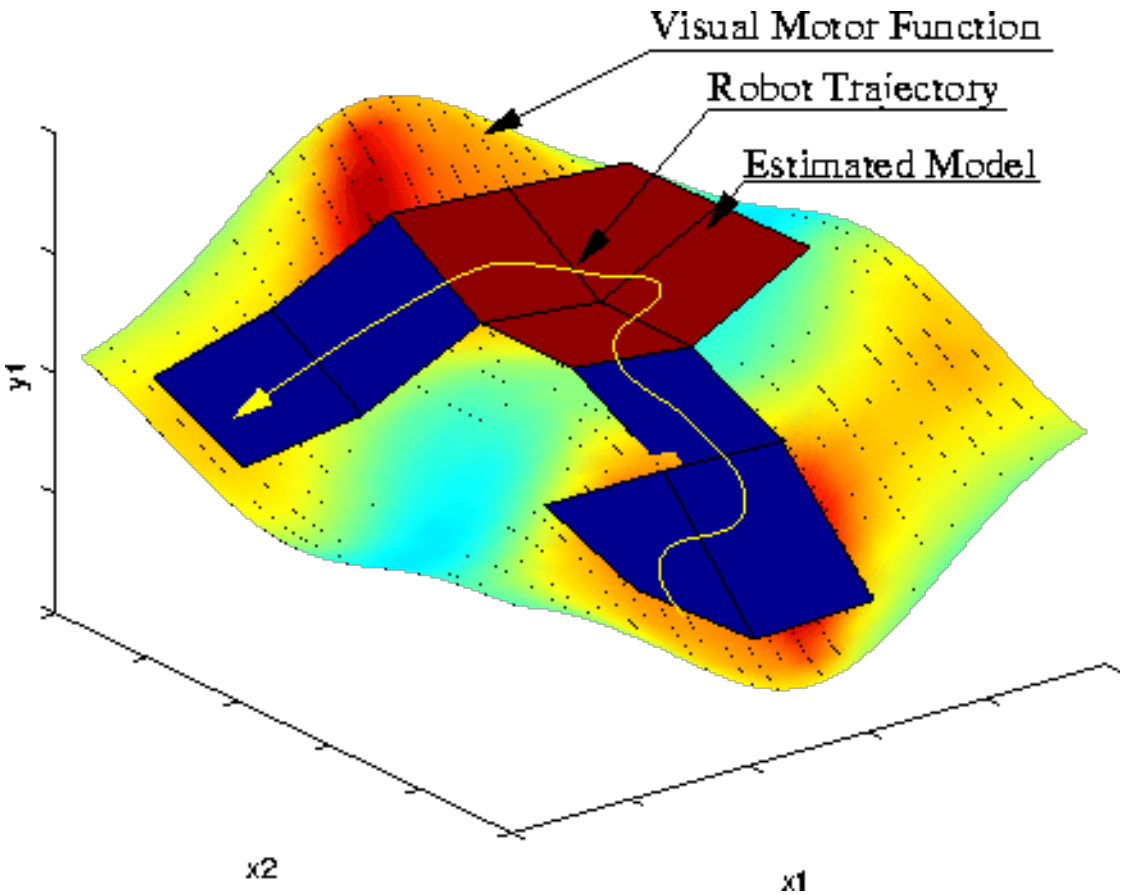
J is $2 \times N$, so use pseudo-inverse

$$\Delta \vec{\theta} = \left(J^\top J \right)^{-1} J^\top \vec{\Delta}$$

But J is only approximate and not true inverse?

So, move $\alpha \Delta \vec{\theta}$ and iterate ($\alpha < 1$)

SERVOING ALGORITHM



see Perceptual Actions: Vision Based Uncalibrated Robot Control by Martin Jägersand

SERVOING ALGORITHM

DO

- Compute $\vec{\Delta}$
- Estimate J
- Compute $\Delta\vec{\theta}$
- Move joints $\alpha\Delta\vec{\theta}$ where $0 < \alpha < 1$

WHILE $\|\vec{\Delta}\| > \tau$ pixels

$\tau > 1$? Maybe robot is 'shakey'

CONVERGENCE?

If α small enough, should always be reducing Δ

As J is linear, moving $\alpha\Delta\vec{\theta}$ should reduce position error by approximately $\alpha\vec{\Delta}$

CONVERGENCE

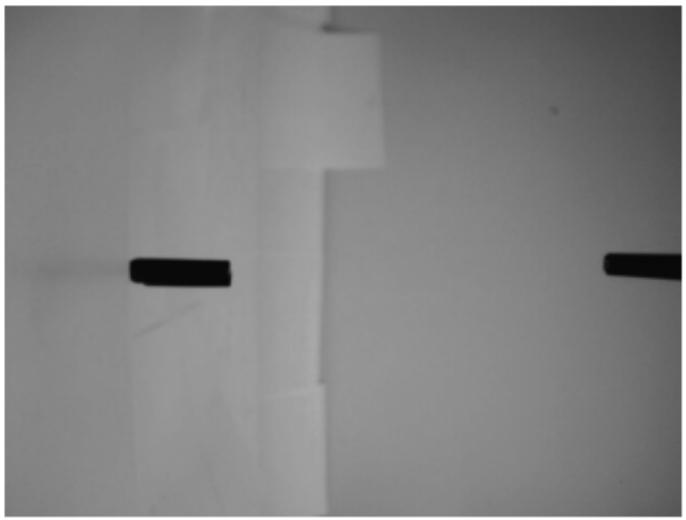
STEP	MOVEMENT	REMAINDER
1	$\alpha\|\vec{\Delta}\|$	$(1-\alpha)\,\|\vec{\Delta}\|$
2	$(1-\alpha)\,\alpha\|\vec{\Delta}\|$	$(1-\alpha)^2\,\|\vec{\Delta}\|$
	$\cdots$	
n	$(1-\alpha)^{n-1}\,\alpha\|\vec{\Delta}\|$	$(1-\alpha)^n\,\|\vec{\Delta}\|$

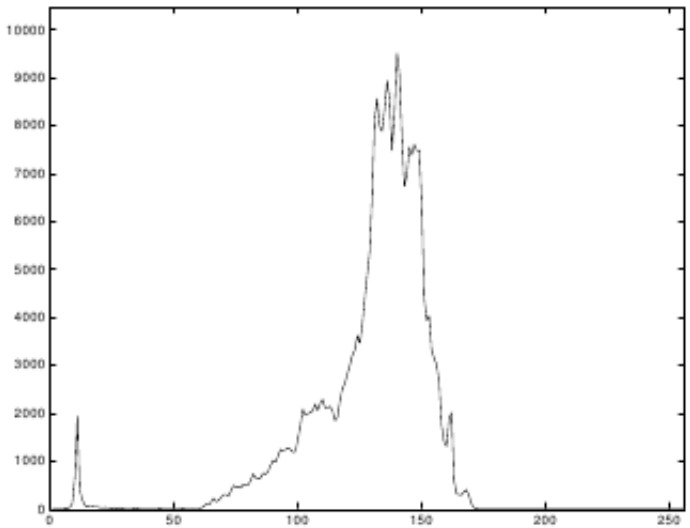
Total movement:

$$\sum_{n=1}^{\infty}(1-\alpha)^{n-1}\,\alpha\|\vec{\Delta}\| = \frac{1}{1-(1-\alpha)}\alpha\|\vec{\Delta}\| = \|\vec{\Delta}\|$$

VISUAL SERVOING RESULTS 1

Initial position and histogram (#pixels per gray level)

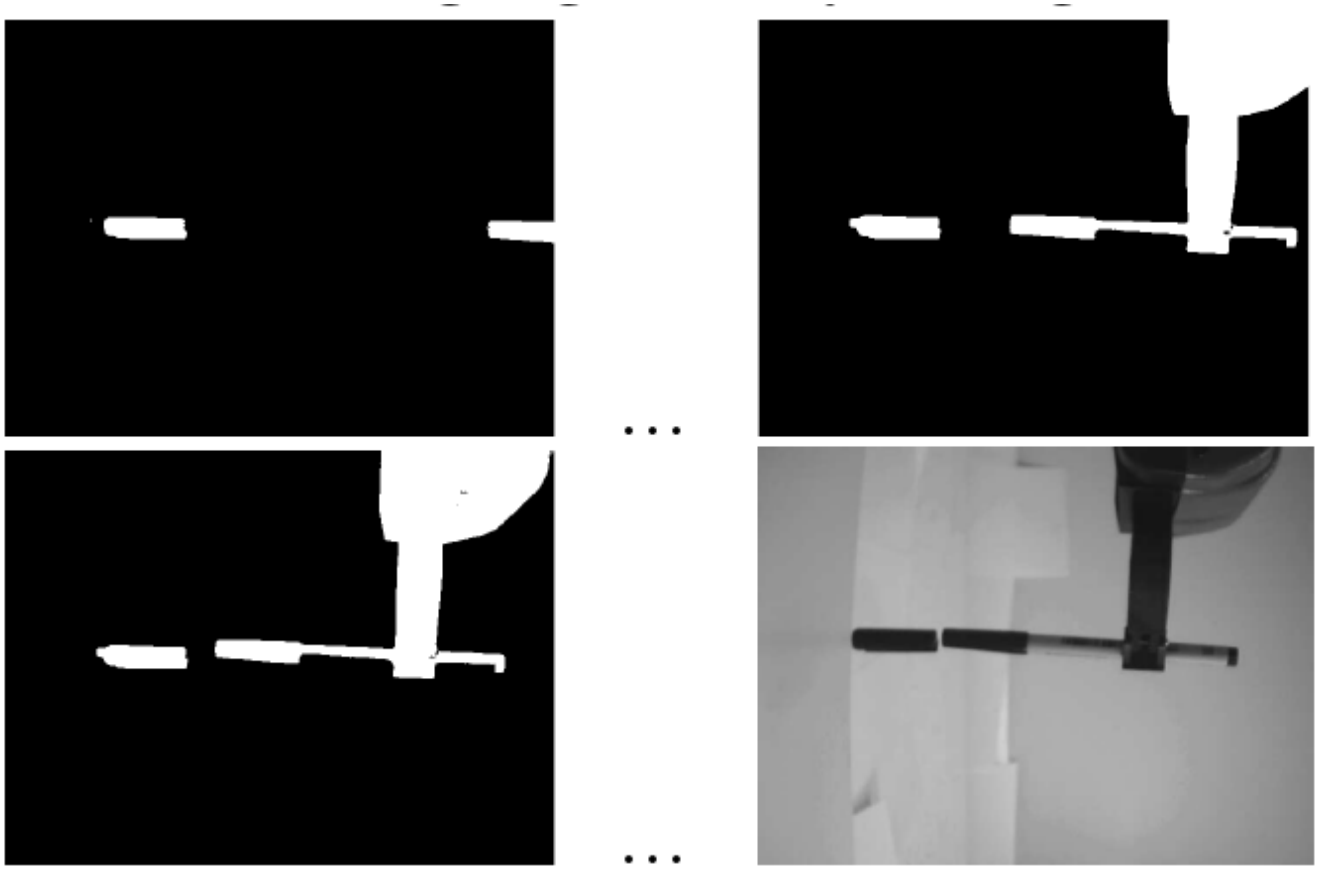




Camera on side

VISUAL SERVOING RESULTS 2

Note: changing binary images



5 iterations

Discussion

- Usually more cameras (image-based visual servoing)
- If the pose of the robot can be estimated based on a robot model a single camera may be sufficient (position-based visual servoing)
- Precision is improved by combination with force feedback
- Camera movements and other sources of errors
- Related problem: Camera positioning; track following in autonomous robots

Summary on visual servoing

- Servoing versus model based control
- Basics of visual servoing
- Linear approximations are usually safe if small steps are taken
- What about efficiency, dynamics, inertia, friction?
  $\Rightarrow$ control theory

Java-based Visual Servo Simulator:
http://www.robot.uji.es/research/projects/javiss

Further reading:

S. Hutchinson, G. D. Hager, P. I. Corke (1996) A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation* **12**:5, 651-670.

IVR: Introduction to Control

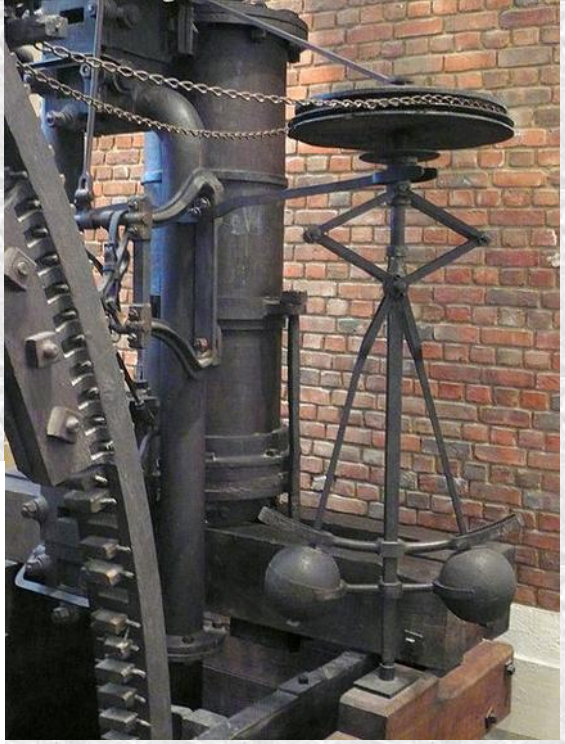
OVERVIEW

- Control systems
- Transformations
- Simple control algorithms

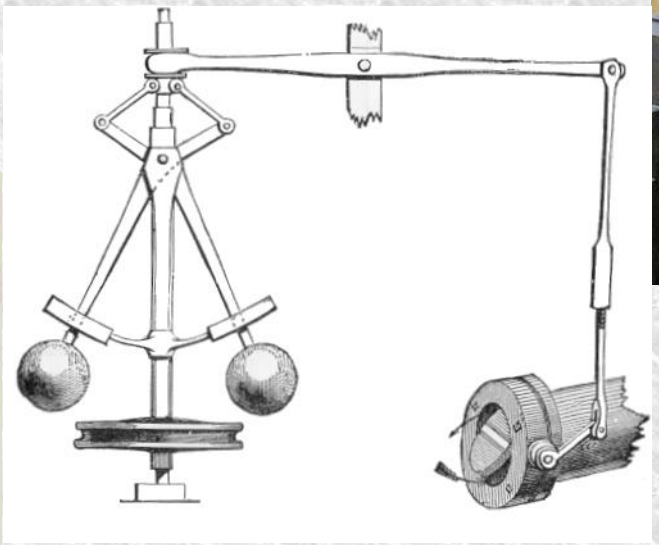
History of control

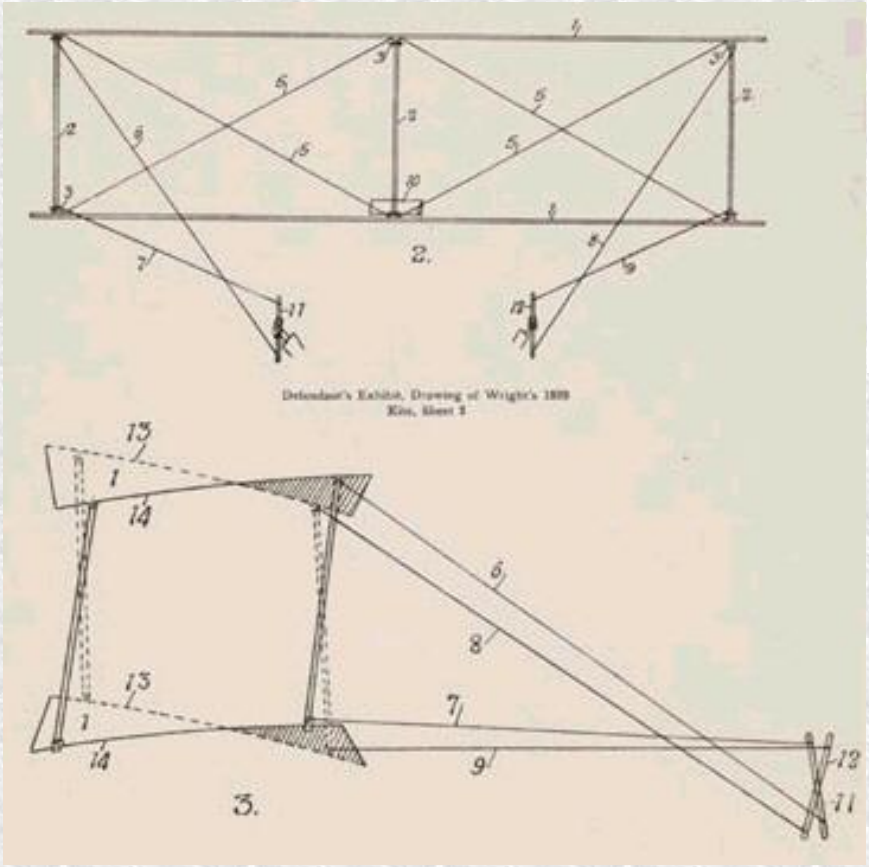
Centrifugal governor

- M. Boulton and J. Watt (1788)
- J. C. Maxwell (1868)
  On Governors.



Science Museum London
(Dr. Mirko Junge)





Pilot control of fixed-wing air-craft: Wright Brothers (1899)
(rather than “inherent stability”)

flight = wings + engines + **control**

Examples of Control

- Aeroplane control
- Cruise control
- Robot control
- Electronics
- Power control
- Thermostat
- Fire control
- Process control
- Space craft control
- Homeostasis & bio-logical motor control
- Control in economy
- ...

Unimate 500 Puma (1983)
Deutsches Museum, Munich
(Theoprakt)

Policeman on Segway PT in Vilnius (Kulmalukko)

Proton Rocket (NASA)

The control problem

How to make a physical system (such as a robot) function in a specified manner?

Particularly when:

- The function would not happen naturally
- The system is subject to a large class of perturbations or changes, e.g.
  - get the mobile robot to a goal
  - keep a walking robot upright
  - move the end-effector to a given position
  - move a camera to track an object

Control

- Dynamical system ("plant")
- Continuous states
- Physical input and output (to/from the system)
- Control actuators
- Controller

Example

- A room (containing air)
- Temperature at certain points in the room
- Heater and measurement device
- A way of switching the heater on or off
- Thermostat

Control

- Dynamical system
- States
- Input and output
- Control actuators
- Controller

Example

- Robot in an environment
- Position and velocity of the robot's DoF
- Sensors and body/ effectors
- Motors, muscles, ...
- Controller hardware/ Control algorithm

Control

- Dynamical system
- States
- Input and output
- Control actuators
- Controller

Example

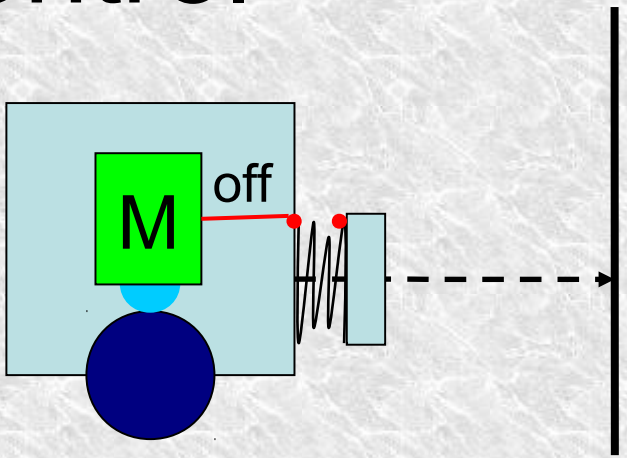
- Khepera robot near a wall
- Distance to the wall
- IR sensors and wheel speeds
- Motors including low-level control
- ???

Questions & Problems

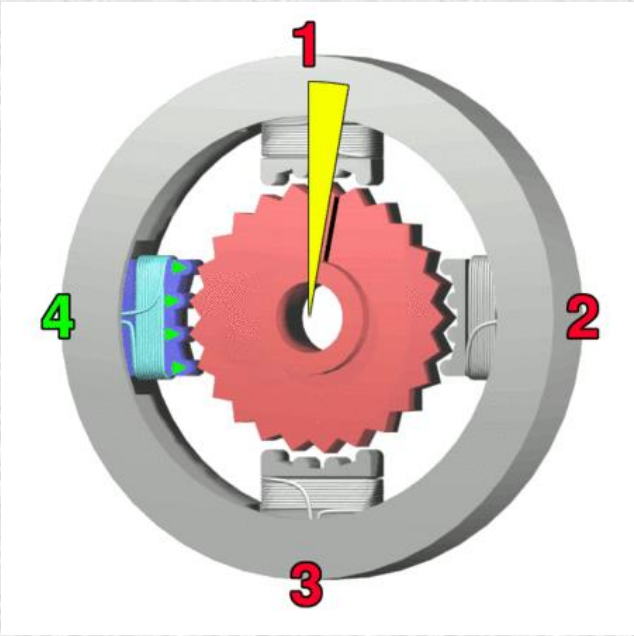
- What control strategy?
- Stability
  - Does the system continue to behave as desired?
- Controllability and observability
  - Are the critical variables accessible and measurable
- Delays
  - Is the measurement up to date, when does the control take effect?
- Efficiency
  - Can the same effect be achieved with less effort?
- Adaptivity
  - Is the control strategy appropriate for changing conditions?

“Bang-bang” control

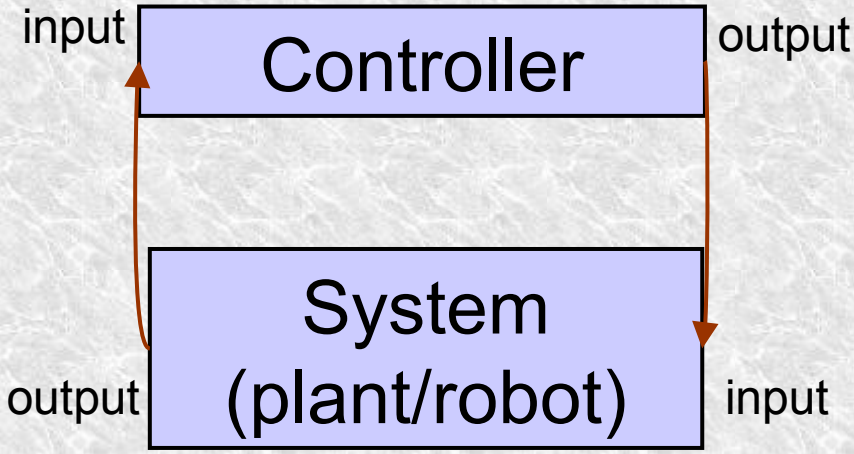
- Simple control method is to have physical end-stop …



- Stepper motor is similar in principle:



Controller



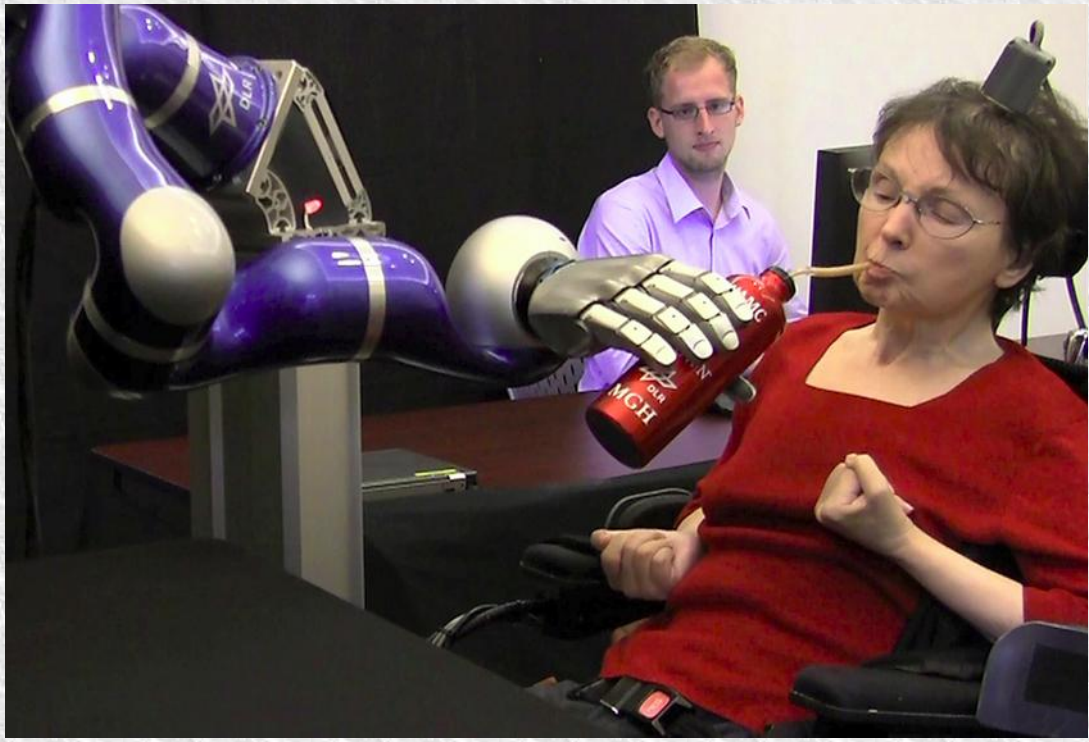
- ‘A device which monitors and affects the operational conditions of a given dynamical system’
- The controller receives the outputs of the controlled system and adjusts the input variables of this system.
- It may also receive signals from a (human) operator or from another controller
- Controllers often aims at affecting the system outputs to stay close to a desired set-point (homeostasis)
- The difference of system output and set-point can serve as feedback telling the controller to what extent the control goal was achieved

Approaches to the control problem

- For a desired outcome, what are the motor commands? ⇒ *Inverse model*
- For given motor commands, what is the outcome? ⇒ *Forward model*
- From observing the outcome, how should we adjust the motor commands to achieve a goal? ⇒ *Feedback control*

Levels of control problem

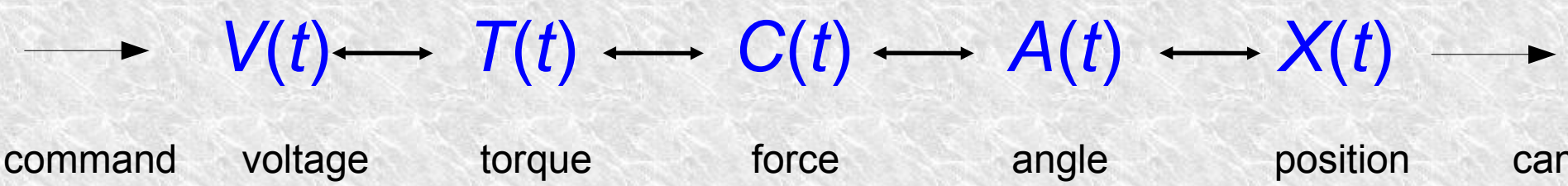
KUKA
robotic arm



Brown University

- Want to move robot hand through set of positions in task space: $X(t)$
- $X(t)$ depends on the joint angles in the arm $A(t)$
- $A(t)$ depends on the coupling forces $C(t)$
- delivered by transmission from motor torques $T(t)$
- $T(t)$ produced by the input voltages $V(t)$

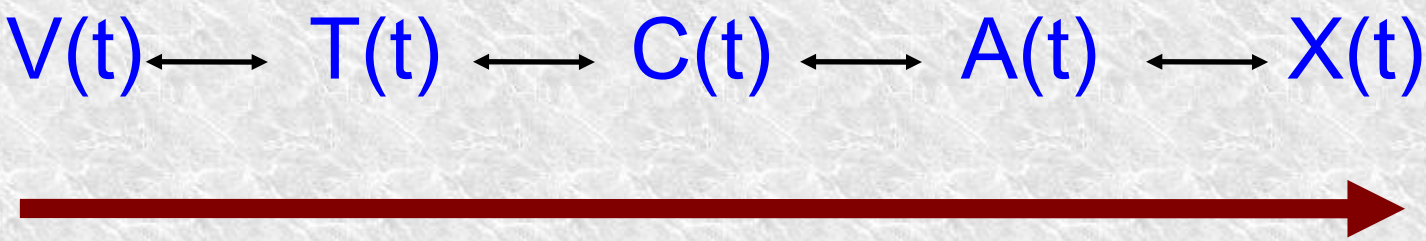
The control system



Depends on:

- Kinematics and geometry: Mathematical description of the relationship between motions of motors and end effector as transformation of coordinates
- Dynamics: Actual motion also depends on forces, such as inertia, friction, etc…

Forward models



- Forward kinematics is not trivial but usually possible
- Forward dynamics is hard and at best will be approximate
- But what we actually need is *backwards* kinematics and dynamics

Difficult!

Inverse models

V(t) ⟷ T(t) ⟷ C(t) ⟷ A(t) ⟷ X(t)

- **Find motor command given desired outcome** (find V given X)
- Solution might not exist
- Ill-posed problems in redundant systems
- Non-linearity of the forward transform
- Robustness, stability, efficiency, ...
- Partial solution and their composition

Summary

- In order to execute a task, robots need information about
  - what actions to perform
  - how to execute the actions
  - the effects on their environment
- This information may be maintained explicitly, (e.g. by a model) or incrementally (in feedback control) or in a combination of both.
- In order to obtain quantitative description of the involved processes, we'll need a systematic approach: Control theory

IVR: Introduction to Control Theory

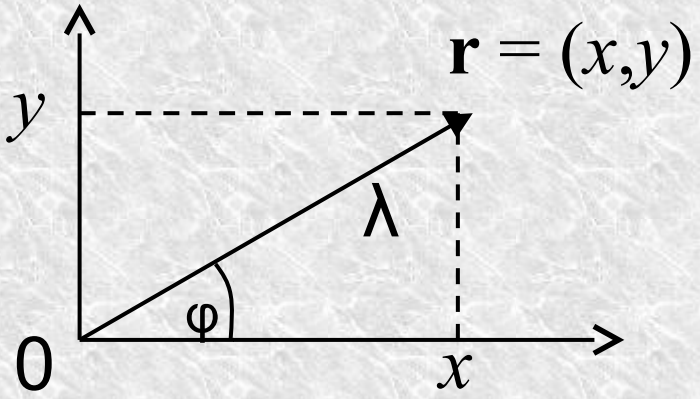
OVERVIEW

- Kinematics
- Differential equations describing robot movements

Kinematic (motion) models

- Differentiating the geometric model provides a motion model (hence sometimes these terms are used interchangeably)
- This may sometimes be a method for obtaining linearity (i.e. by looking at position change in the limit of very small changes)

Example:
A simple arm model



$x = \lambda \cos \varphi$

$y = \lambda \sin \varphi$

$\varphi = \text{atan2}(x,y)$

e.g. vertical movement:

$$y(t) = \lambda \sin \varphi(t)$$

$$\frac{dy}{dt} = \lambda \cos \varphi(t) \frac{d\varphi}{dt}$$

Dynamic (kinetic) models

- Kinematic models neglect forces: motor torques, inertia, gravity, friction, …
- Forces arise according to Newtons 2nd law: $F = m\,a$
- $F = m\,a = m\,\mathrm{d}^2y/\mathrm{d}t^2$ (need to know mass!)

$$y(t) = \lambda \sin \varphi(t)$$

$$\frac{dy}{dt} = \lambda \cos \varphi(t) \frac{d\varphi}{dt}$$

$$\frac{d^2y}{dt^2} = -\lambda \sin \varphi(t) (d\varphi/dt)^2 + \lambda \cos \varphi(t)\, d^2\varphi/dt^2$$

- To control a system, we need to understand the continuous process

Dynamical systems

- Differ from standard computational view on systems:
  - **Perception-action loop** i.e. the influence of outputs onto inputs is taken into account (instead of: input → processing → output)
  - Set of all possible states forms a *state-space* (can be continuous or discrete)
  - Behaviour is a trajectory in the state space
- Not only physical systems: On-going debate whether human cognition is better described as computation or as a dynamical system (e.g. van Gelder, 1998). Similarly: society, economy, ecosystems etc.

Differential Equations

Using known relations between quantities and their rate of change in order to find out how these quantities change

- Mathematics:
  Equation that is to be solved for an unknown function
- Physics:
  Description of processes in nature
- Engineering:
  Realizability of a goal by a plant by including control terms
- Informatics:
  Tool for realistic modeling

$$\frac{dx(t)}{dt} = f(x(t))$$

$$x(t_0) = x_0$$

Solving a simple diff-equation

$$\dot{x}(t) = a \cdot x(t)$$

$$\frac{dx(t)}{dt} = a \cdot x(t)$$

$$(x(t) \neq 0)$$

$$\frac{1}{x(t)} \frac{dx(t)}{dt} = a$$

$$\int_{t_0}^{t} \frac{1}{x(t')} \frac{dx(t')}{dt'} dt' = \int_{t_0}^{t} a \, dt'$$

$$\int_{x_0 = x(t_0)}^{x} \frac{1}{x'} dx' = \int_{t_0}^{t} a \, dt'$$

$$\log x' \Big|_{x_0}^{x} = a t' \Big|_{t_0}^{t}$$

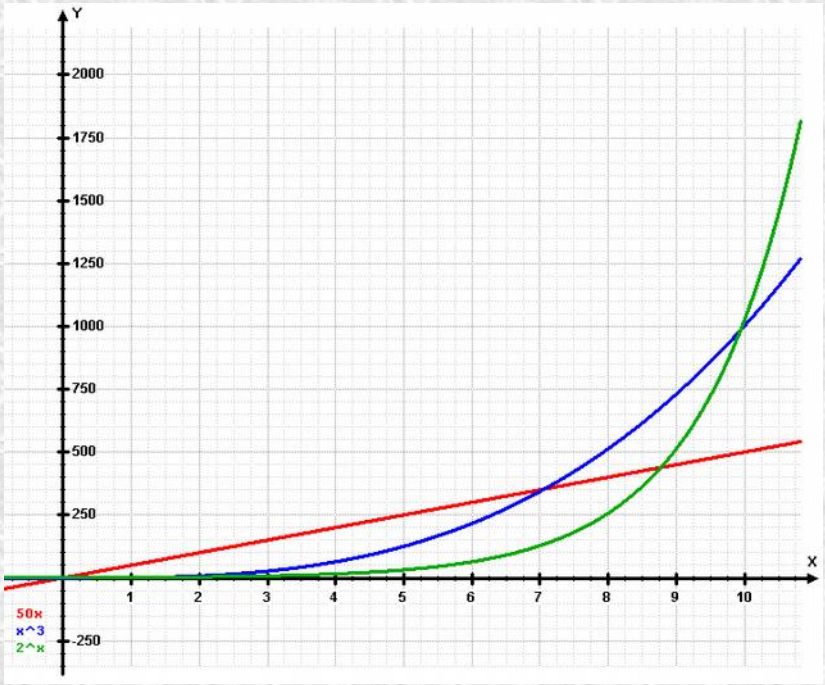
$$\log x - \log x_0 = at - at_0$$

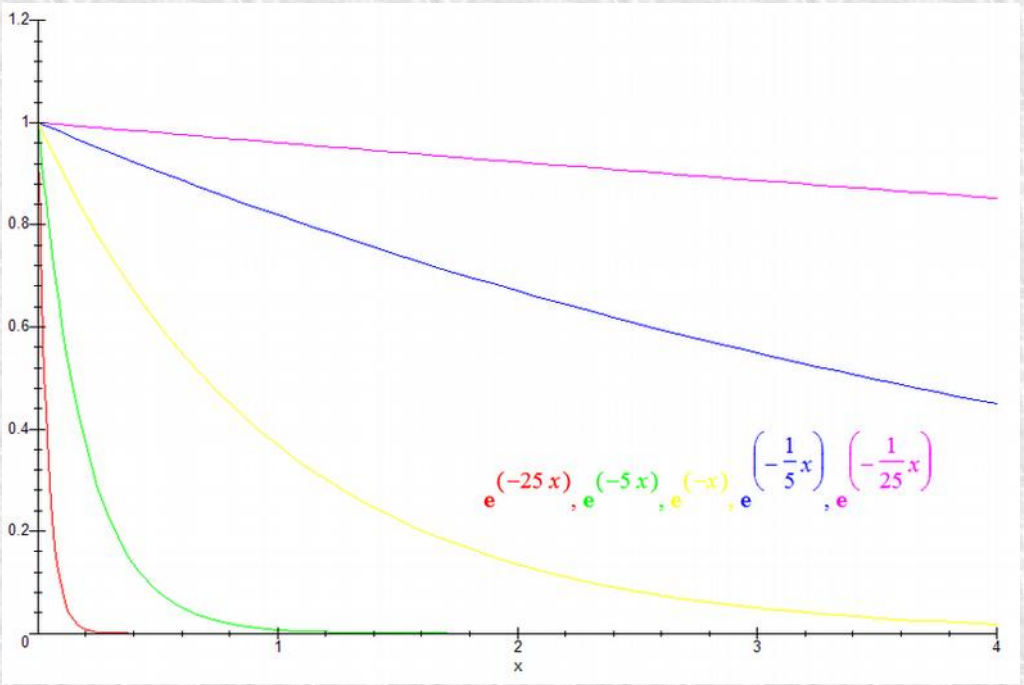
$$\log x(t) = \log x_0 + a(t - t_0)$$

$$x(t) = x_0 \cdot \exp a(t - t_0)$$

Differential Equations

$$\dot{x}(t) = a \cdot x(t) \quad \Longrightarrow \quad x(t) = x_0 \cdot \exp a(t - t_0)$$





$a > 0$ increases faster than any linear or polynomial function [for $x(0) > 0$]

$a < 0$ decay with time scale: $-1/a = \tau$

Problem: Robots are non-linear

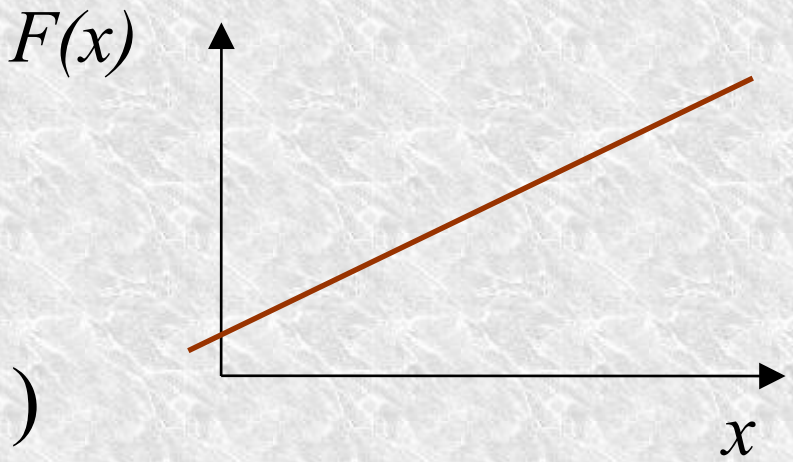
Linear differential equations (like the previous one) contain only linear combinations of variable function(s) and derivatives

- Reminder:

  Linear function:

$$F(x) = a + bx$$

$$F(x_1 + x_2) = F(x_1) + F(x_2)$$



- Linear combinations of solutions of the same linear equation are again solutions
  (in the previous differential equation different solutions differ only by x_0)
- For linear systems there are good formal methods
- However, dynamics of most robots is non-linear

Summary

- Whatever happens can be described as a dynamical system
- Differential equations are the native language of continuous dynamical systems
- Control systems describe the behaviour of robots
- For small step sizes, non-linear behaviour can often be linearly approximated
- In some cases we can use discretised form of the equations

Overview and Outlook

- Inverse and forward models (last time)
  describe transformations (flow of information) in a control system. Feedback control is less dependent on models as it uses plant outputs to determine control actions
- Modes of control (next lecture)
  - Open loop (based on an inverse model)
  - Feed-forward (open loop, but reacting pre-emptively to known disturbances)
  - Closed loop (feedback control)
- Feedback control (to be discussed later)
  - requires less domain knowledge
  - can use forward models to reduce delay effects

IVR: Characterisation of a Control System

M. Herrmann

Overview

- Modelling an control system
- Stationary behaviour and time constant of control systems
- Model identification

Differential Equations

$$\frac{dx(t)}{dt} = f(x(t)) \quad \text{subject to } x(t_0) = x_0$$

- Example from last lecture:

$$\frac{dx(t)}{dt} = a\,x(t) \quad \text{subject to } x(t_0) = x_0$$

- Solution

$$x(t) = x_0 \exp(a(t - t_0))$$

- $a > 0$ fast increase for $x_0 > 0$
- $a < 0$ decay with time constant $\tau := -\frac{1}{a}$

Example: Electric motor

- Ohm's law & Kirchhoff's law

$$V_B = IR + e$$

- Motor generates voltage $e = k_1 s$
- Vehicle acceleration

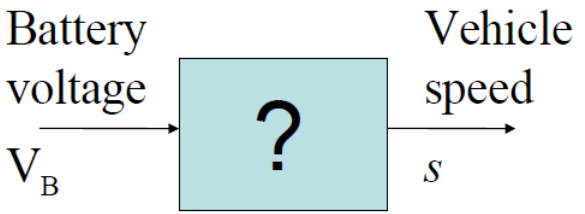
$$\frac{ds}{dt} = \frac{\tau}{M}$$

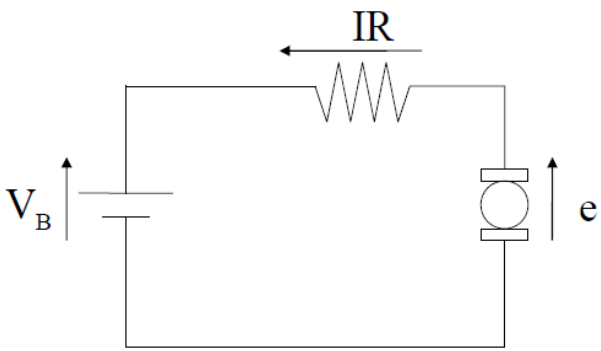
- Torque τ is proportional to current:

$$\tau = k_2 I$$

- Putting together:

$$V_B = \frac{M}{k_2} R \frac{ds}{dt} + k_1 s$$





Solving differential equations numerically

1. Given process model

$$V_B = k_1 s + \frac{M}{k_2} R \frac{ds}{dt}$$

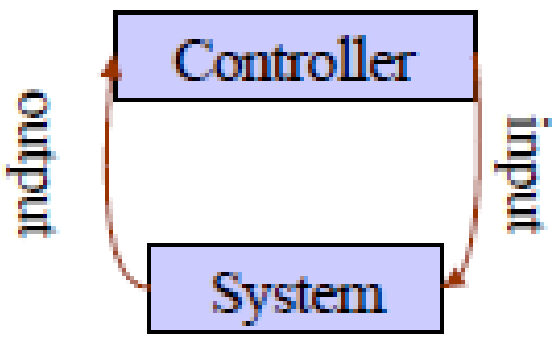
   Rewrite

$$\frac{ds}{dt} = -\frac{k_1 k_2}{RM} s + \frac{k_2}{RM} V_B$$

2. Choose a small step size Δt
3. Start at $t = 0$ and $s(0) = s_0$
4. Iterate $s(t + \Delta t) = s(t) + \Delta t \frac{ds}{dt}$ until $t = T$
5. How do control $s(t)$ towards a desired value by changing V_B?

Abstract form as a control system

$C = As + B\frac{ds}{dt}$	control system	plant and controller
$A + B\frac{d}{dt}$	process dynamics	operator applied to state
s	state variable	output: plant $\rightarrow$ controller
C	control variable	input: controller $\rightarrow$ plant



$$C = As + B\frac{ds}{dt}$$

$$z = As - C \implies z = -B\frac{ds}{dt}$$

$$\frac{dz}{dt} = A\frac{ds}{dt} \implies z = -\frac{B}{A}\frac{dz}{dt}$$

$$z(t) = z_0 e^{-\frac{A}{B}(t-t_0)} \overset{z=As-C}{\Longrightarrow} s(t) = s_0 e^{-\frac{A}{B}(t-t_0)} + \frac{C}{A}\left(1 - e^{-\frac{A}{B}(t-t_0)}\right)$$

Stationary state: $\frac{C}{A}$; time scale (decay by a factor of $\frac{1}{e}$): $\tau = \frac{B}{A}$

Back to the Example

$$V_B = k_1 s + \frac{M}{k_2} R \frac{ds}{dt}$$

$C = As + B\frac{ds}{dt}$ with $A = k_1$, $B = \frac{M}{k_2}R$, $C = V_B$

Stationary behaviour $s\,(t \to \infty) = \frac{C}{A}$

$$\frac{ds}{dt} = 0 \implies V_B = k_1 s \implies s\,(t \to \infty) = \frac{V_B}{k_1}$$

Time scale (decay by a factor of $\frac{1}{e}$)

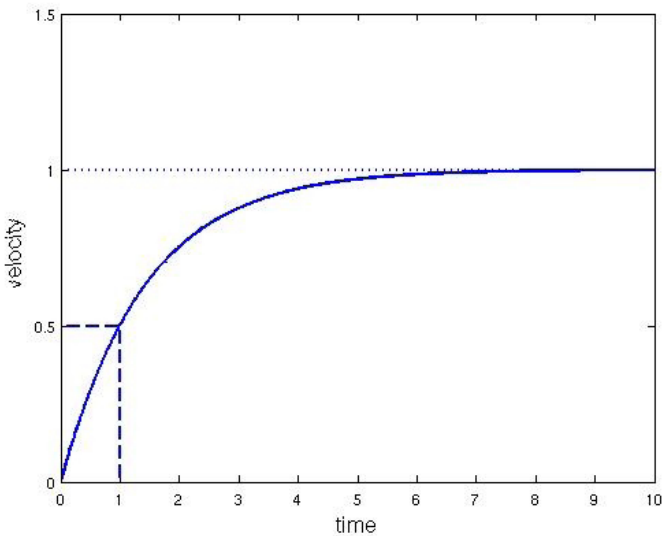
$$\tau = \frac{B}{A} = \frac{MR}{k_1 k_2}$$

Half-life of decay

Decay by a factor of $\frac{1}{2}$

$$s(t) = s_0 \mathrm{e}^{-\frac{k_1 k_2}{MR}(t-t_0)} + \frac{V_B}{k_1}\left(1 - \mathrm{e}^{-\frac{k_1 k_2}{MR}(t-t_0)}\right)$$

Starting from previous state $s_0 = 0$ towards new stationary behaviour at $s = \frac{V_B}{k_1}$



$$s(t) = \frac{V_B}{k_1}\left(1 - \mathrm{e}^{-\frac{k_1 k_2}{MR}(t-t_0)}\right)$$

$$\left(\frac{1}{2}\right)\frac{V_B}{k_1} = \frac{V_B}{k_1}\left(1 - \mathrm{e}^{-\frac{k_1 k_2}{MR}(t-t_0)}\right)$$

$$\frac{1}{2} = \mathrm{e}^{-\frac{k_1 k_2}{MR}(t-t_0)}$$

$$t_{\mathrm{half}} = -\ln\left(\frac{1}{2}\right)\frac{MR}{k_1 k_2} \approx 0.7\frac{MR}{k_1 k_2}$$

Example

Suppose $k_1 = 7$, $\frac{MR}{k_2} = 20$,

$$V_B = 7s + 20\frac{ds}{dt}$$

If the robot starts at rest and 7V are applied then steady state speed is

$$s = \frac{V_B}{k_1} = 1\text{m sec}^{-1}$$

$$\tau_{\frac{1}{2}} \approx 0.7\frac{MR}{k_1 k_2} = 0.7\frac{20}{7} = 2\text{sec}$$

Time taken to cover half the gap between current and steady-state speed.

Identification of a control problem

If $K = 5$, and $s_\infty = 8$, and the system takes 14 seconds to reach $s = 6$ starting from $s = 0$. What is the process equation?

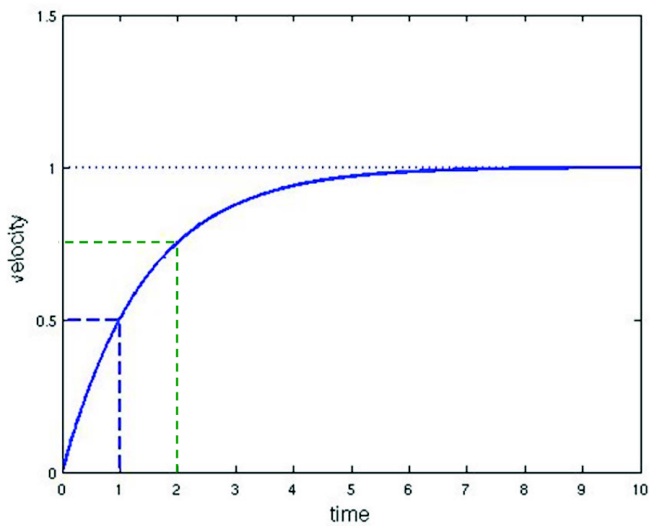
- Determine A and B in

$$Ks_\infty = B\frac{ds}{dt} + As$$

- General solution of the equation:

$$\tau = \frac{B}{A} \qquad s_\infty = \frac{C}{A}$$

where $C = Ks_\infty = 40$



Solution of the process equation (unit scale)

- Thus $A = \frac{C}{s_\infty} = K = 5, \quad B = \tau A$
- How do we obtain τ? (Hint $\tau = 7s$, i.e. $B = 35$)

Controlling the motor over time

Process model

$$V_B = k_1 s + \frac{M}{k_2} R \frac{ds}{dt}$$

Stationary behaviour (steady state)

$$s\left(t \to \infty\right) = \frac{V_B}{k_1}$$

Inverse model:

$$V_B = k_1 \, s_{\text{goal}}$$

Solution provides forward model:

$$s\left(t\right) = s_0 \mathrm{e}^{-\frac{MR}{k_1 k_2}\left(t - t_0\right)} + \frac{V_B}{k_1}\left(1 - \mathrm{e}^{-\frac{MR}{k_1 k_2}\left(t - t_0\right)}\right)$$

Summary

- In order to derive a process model, we need to understand the physics of the system
- For simple processes, the system can be characterised by stationary state and the time constant of the approach towards this state
- Given the model equation, a few measurements can help to derive the explicit model equation and thus to obtain the trajectory of the system

IVR: Open- and Closed-Loop Control

M. Herrmann

informatics

Overview

- Open-loop control
- Feed-forward control
- Towards feedback control

Controlling the motor over time

Process model

$$V_B = k_1 s + \frac{M}{k_2} R \frac{ds}{dt}$$

Stationary behaviour (steady state)

$$s\left(t \to \infty\right) = \frac{V_B}{k_1}$$

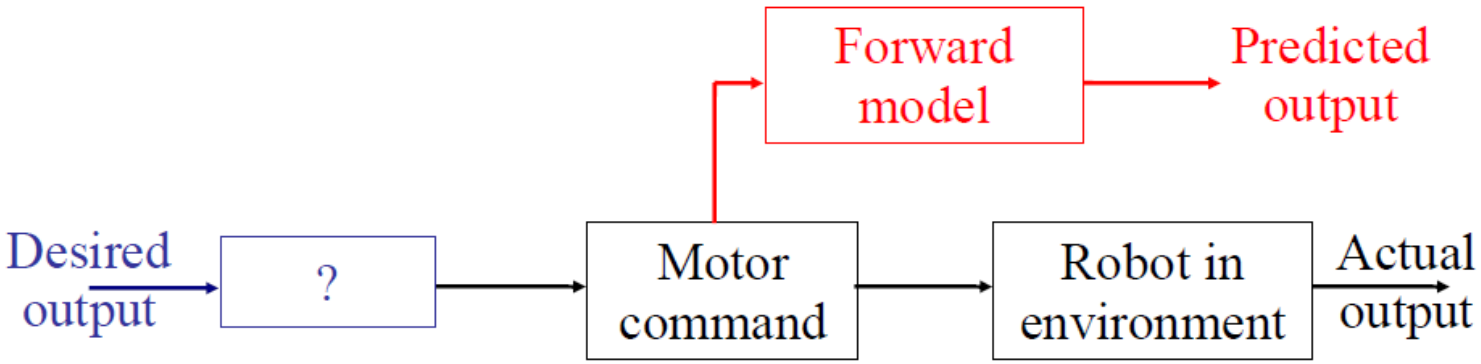
Inverse model:

$$V_B = k_1 \, s_{\text{goal}}$$

Solution provides forward model:

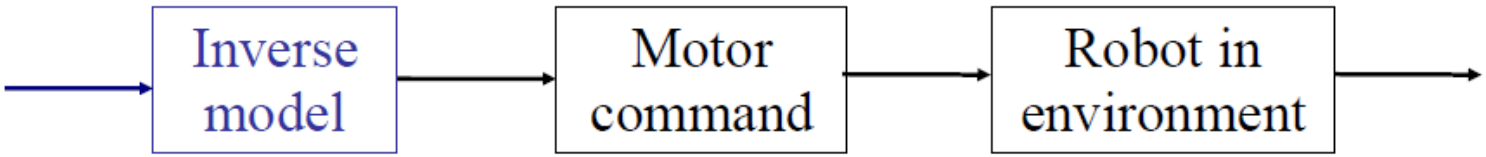
$$s\left(t\right) = s_0 \mathrm{e}^{-\frac{MR}{k_1 k_2}\left(t - t_0\right)} + \frac{V_B}{k_1}\left(1 - \mathrm{e}^{-\frac{MR}{k_1 k_2}\left(t - t_0\right)}\right)$$

The control problem



- Forward: Given the control signals, can we predict the motion of the robot?
- Inverse: Given the desired motion of the robot: Can we determine the right control signals?
- Control theory aims at unique or easy exact solutions
- Difficult in the presence of non-linearities or noise

Open loop control



Examples:

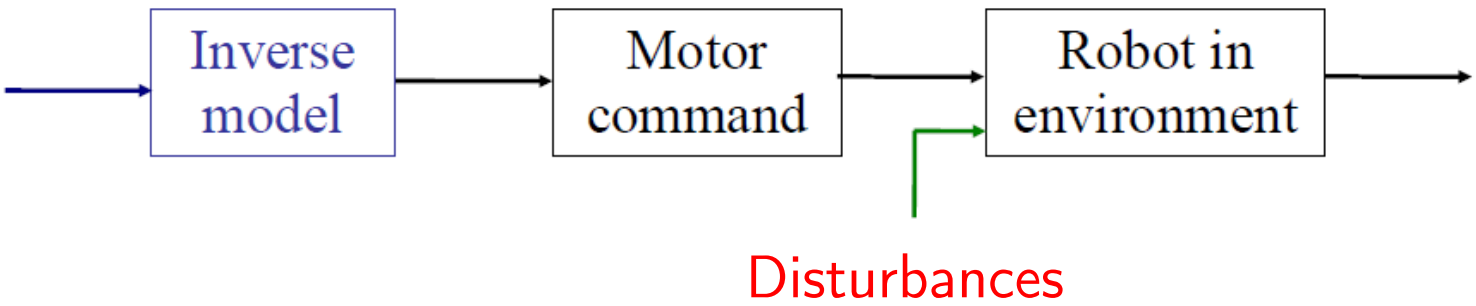
- To execute a memorised trajectory, produce appropriate sequence of motor torques
- To obtain a goal, make a plan and execute it
  - means-ends reasoning could be seen as inverting a forward model of cause-effect
- 'Ballistic' movements such as saccades

Open loop control

- Potentially cheap and simple to implement e.g. if solution is already known.
- Fast, e.g. useful if feedback would come too late.
- Benefits from calibration e.g. tune parameters of approximate model.
- If model unknown, may be able to use statistical learning methods to find a good fit e.g. using neural networks

Open loop control

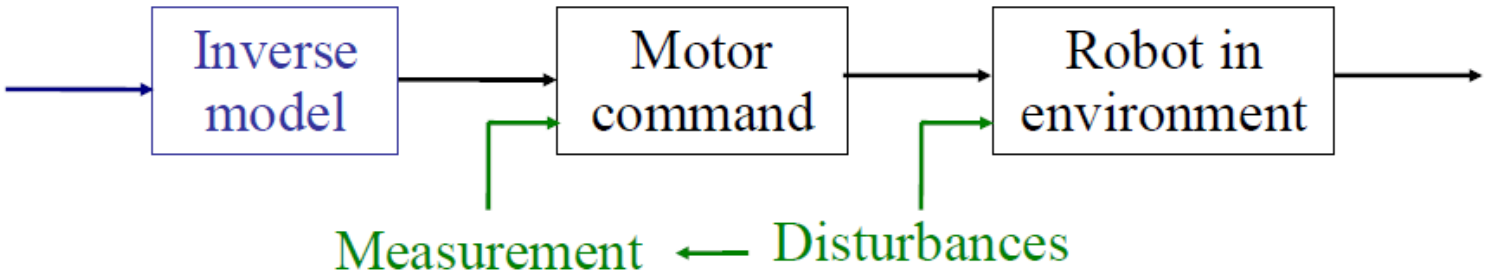
Neglects possibility of disturbances, which might affect the outcome.



For example:

- Change in temperature may change the friction in all the robot joints.
- Unexpected obstacle may interrupt trajectory

Feed-forward control



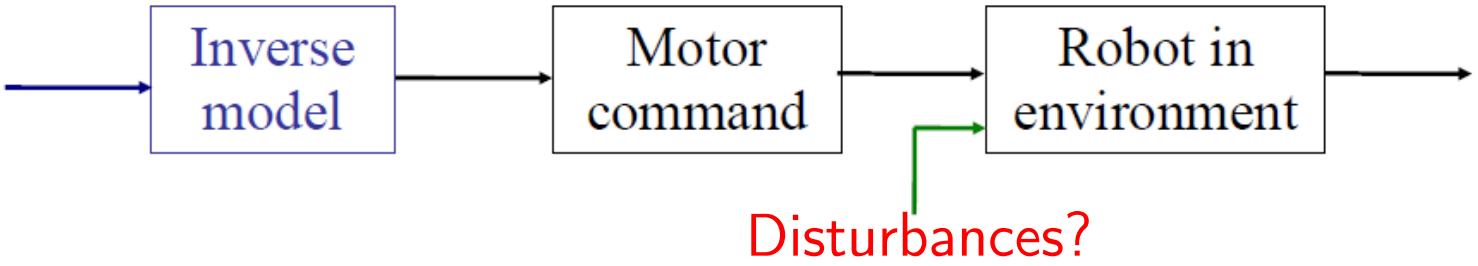
- One solution is to measure the (potential) disturbance and use this to adjust the control signals.
- For example
  - Thermometer signal alters friction parameter.
  - Obstacle detection produces alternative trajectory.

Feed-forward control

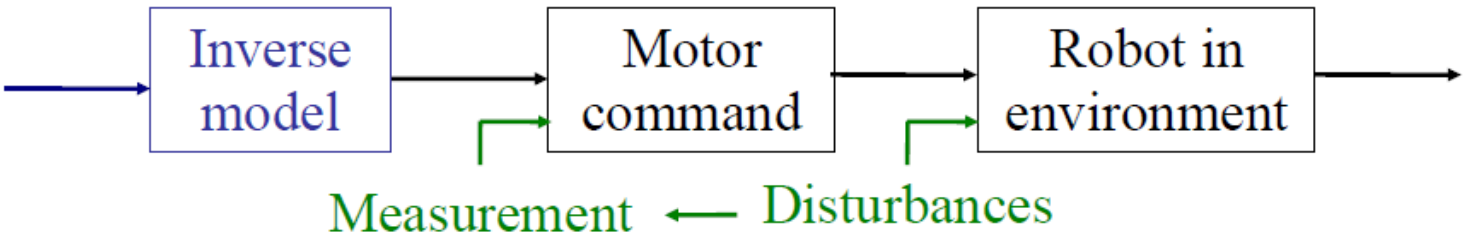
- Can sometimes be effective and efficient.
- Requires anticipation, not just of the robot process characteristics, but of possible changes in the world.
- Does not provide or use knowledge of actual output (for this need to use **feedback** control).

Control paradigms

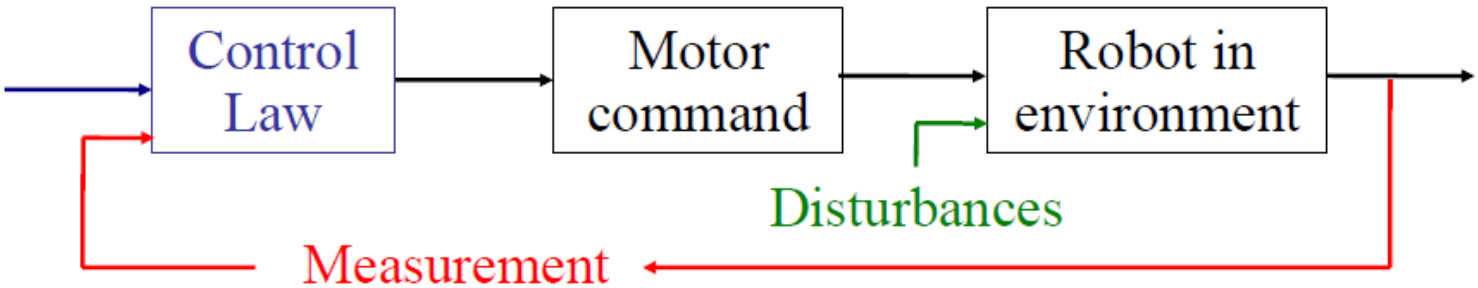
Open loop control



Feed-forward control



Feedback control

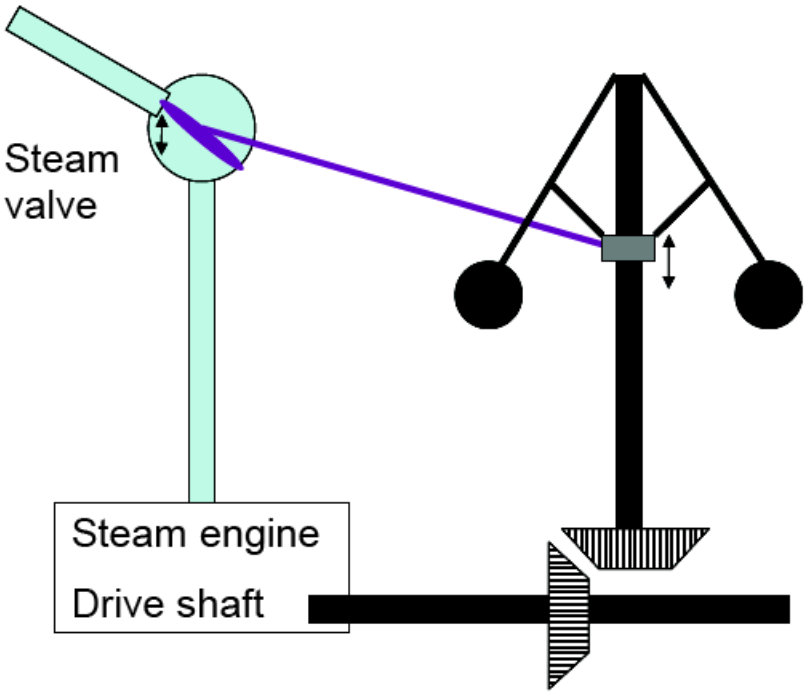


Combining control methods

- In practise most robot systems require a combination of control methods
  - Robot arm using servos on each joint to obtain angles required by geometric inverse model
  - Forward model predicts feedback, so can use in fast control loop to avoid problems of delay
  - Feed-forward: Measurements of disturbances can be used to adjust feedback control parameters
  - Training of an open-loop system is essentially a feedback process

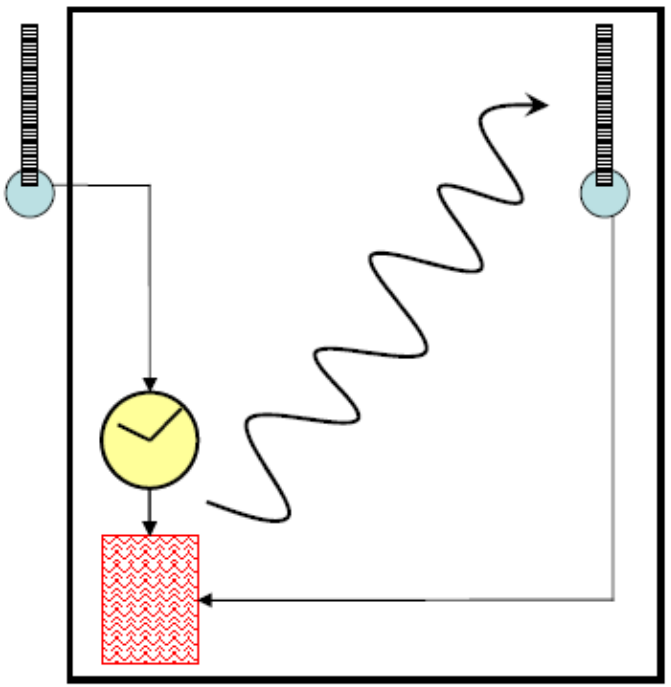
Feedback control

Example: Watt governor

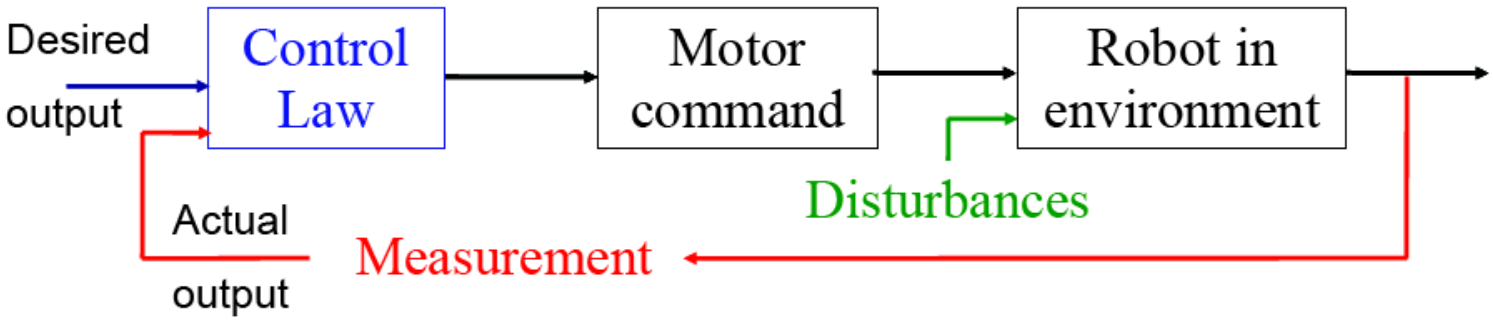


Compare: Room heating

Open loop: For desired temperature, switch heater on, and after pre-set time, switch off.
Feed forward: Use thermometer outside room to compensate timer for external temperature.
Feedback: Use thermometer inside room to switch off when desired temperature reached.
NO PREDICTION REQUIRED!



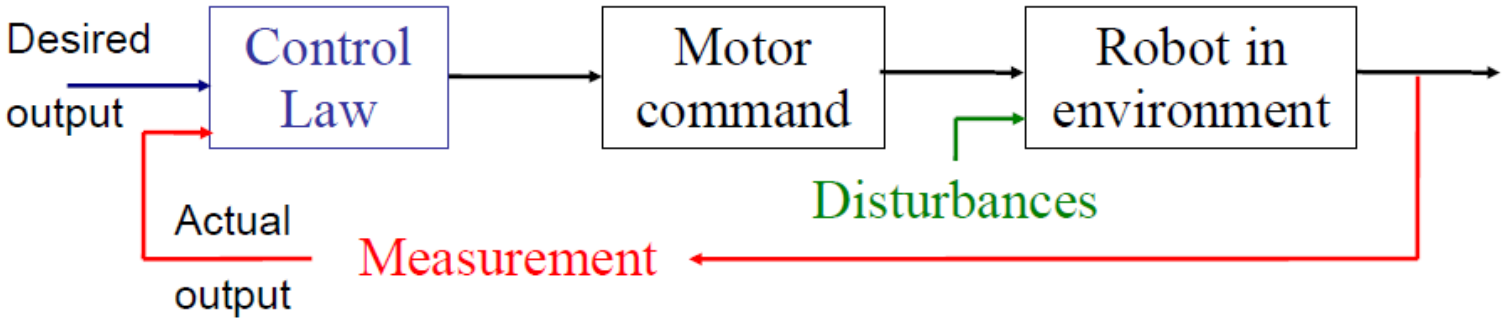
Control laws in feed-back control



On-off: Switch system if desired = actual

Servo: control signal proportional to difference between desired and actual, e.g. spring:

Complex control law



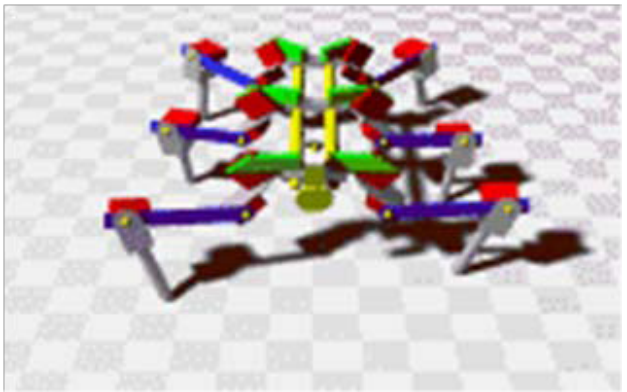
- May involve multiple inputs and outputs
- E.g. 'homeostatic' control of human body temperature
  - Multiple temperature sensors linked to hypothalamus in brain
  - Depending on difference from desired temperature, regulates sweating, vasodilation, piloerection, shivering, metabolic rate, behaviour...
  - Result is reliable core temperature control

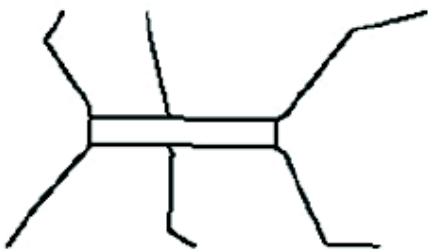
Negative vs. Positive Feedback

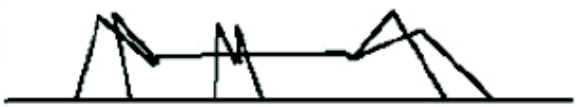
- Examples so far have been negative feedback: control law involves subtracting the measured output, acting to decrease difference.
- Positive feedback results in amplification:
  - e.g. microphone picking up its own output
  - e.g. 'runaway' selection in evolution
- Generally taken to be undesirable in robot control, but sometimes can be effective
  - e.g. in exploratory control, self-excitation
  - model of stick insect walking (H. Cruse et al., 1995)
  - to counteract negative feedback from the environment

Stick insect walking

- 6-legged robot has 18 degrees of mobility.
- Difficult to derive co-ordinated control laws
- But have linked system: movement of one joint causes appropriate change to all other joints.
- Can use signal in a feed-forward loop to control 12 joints (remaining 6 use feedback to resist gravity).







Advantages of feedback control

- Major advantage is that (at least in theory) feedback control does not require a model of the process.
  - E.g. thermostats work without any knowledge of the dynamics of room heating (except for time scales and gains)
- Thus controller can be very simple and direct
  - E.g. hardware governor does not even need to do explicit measurement, representation and comparison
- Can (potentially) produce robust output in the face of unknown and unpredictable disturbances
  - E.g. homeostatic body temperature control keeps working in unnatural environments

Issues to be solved

- Requires sensors capable of measuring output
  - Not required by open-loop or feed-forward
- Low gain is slow, high gain is unstable
- Delays in feedback loop will produce oscillations (or worse)
- In practise, need to understand process to obtain good control:
  - E.g. James Clerk Maxwell's analysis of dynamics of the Watt governor

IVR: Feedback control
Proportional and Proportional Integral Control

M. Herrmann

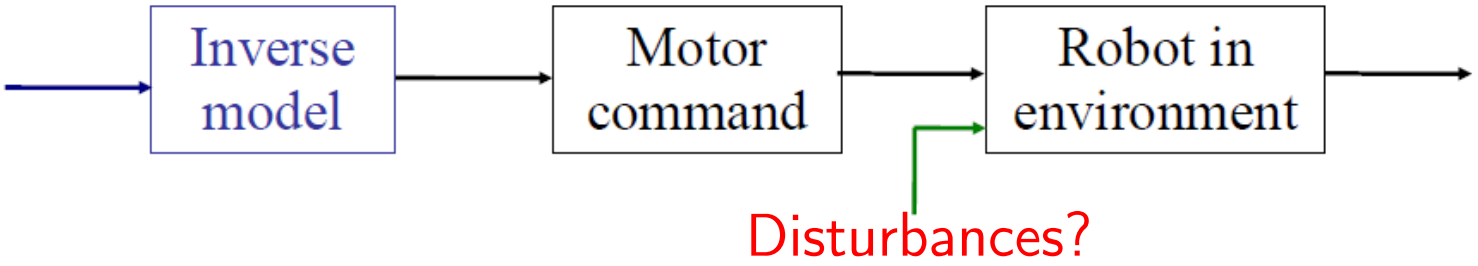
informatics

Overview

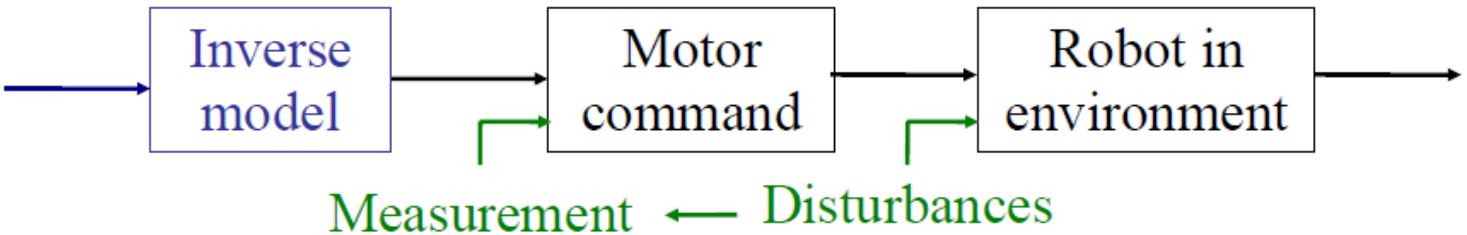
- Feedback control
- Proportional error control
- Proportional Integral control
- Proportional Derivative control
- Next time PID

Control paradigms

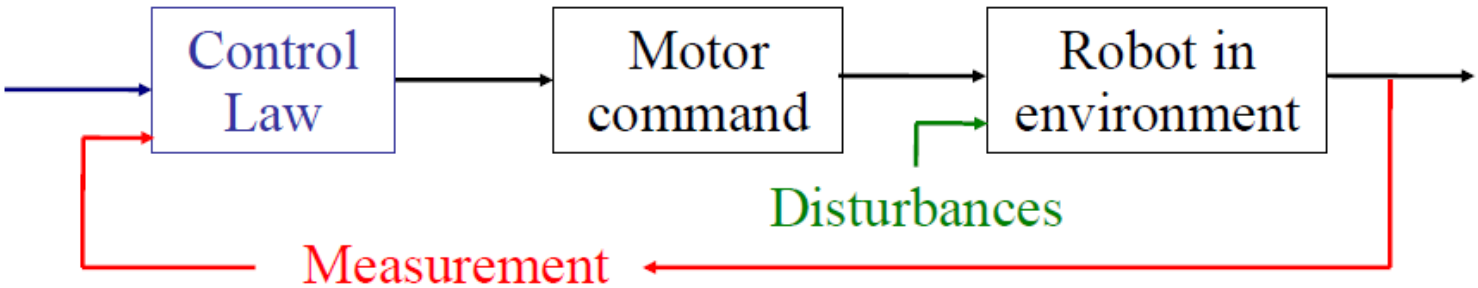
Open loop control



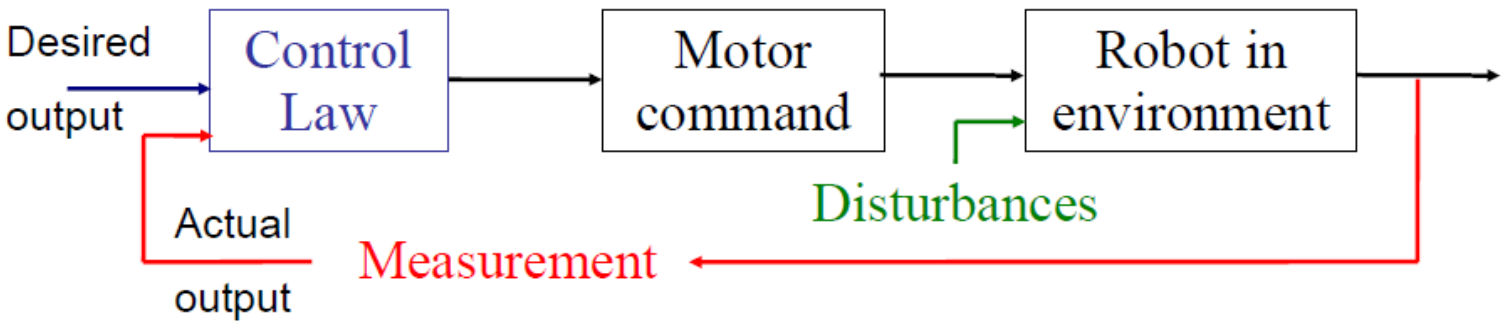
Feed-forward control



Feedback control



Feedback control

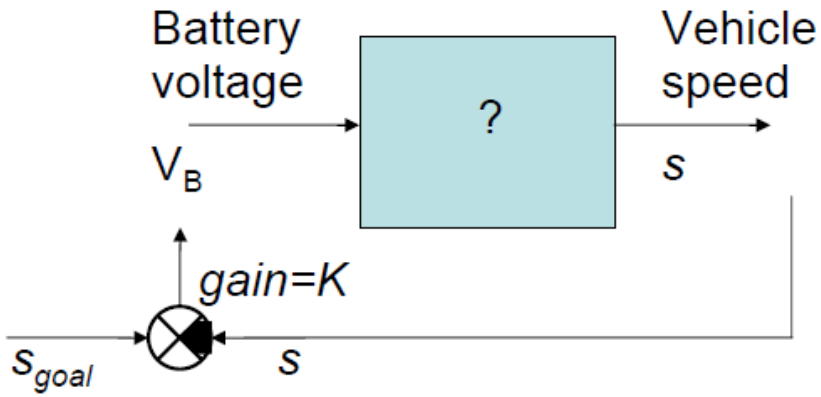


- Simple and direct, does not require a model of the process
- Robust in the face of unknown and unpredictable disturbances
- Requires sensors capable of measuring output
- Tuning required: Low gain is slow, high gain is unstable
- Delays in feedback loop may interfere with the control law

Servo control (Proportional error control)

Simple dynamic example ($C = V_B, B = \frac{M}{k_2}R$, $A = k_1$):

$$V_B = \frac{M}{k_2}R\frac{ds}{dt} + k_1 s$$



Control law:

$$V_B = K\left(s_{\text{goal}} - s\right)$$

So now have new process:

$$K\left(s_{\text{goal}} - s\right) = \frac{M}{k_2}R\frac{ds}{dt} + k_1 s$$

$$Ks_{\text{goal}} = \frac{M}{k_2}R\frac{ds}{dt} + (K + k_1)\, s$$

With steady state:

$$s_\infty = \frac{Ks_{\text{goal}}}{K + k_1}$$

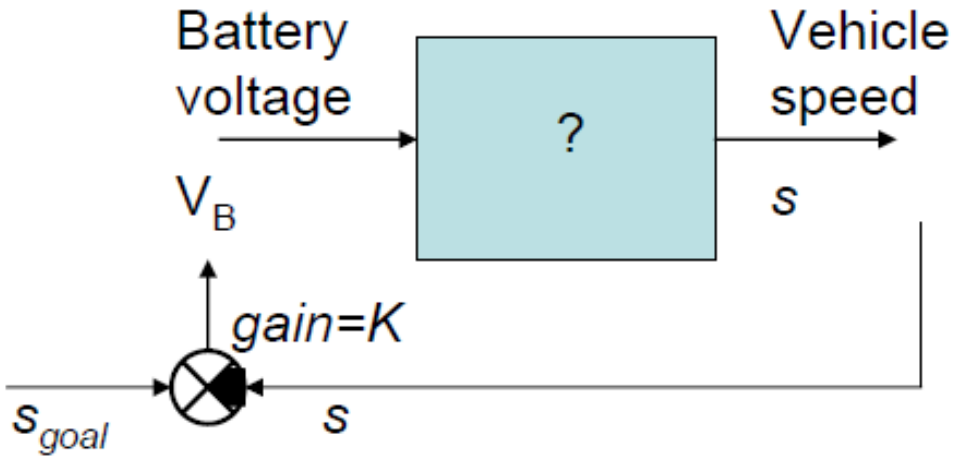
And half-life:

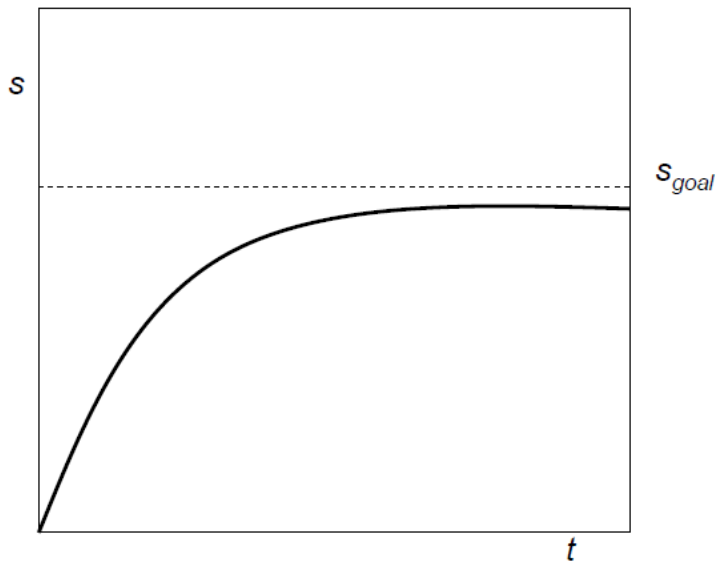
$$\tau_{\frac{1}{2}} = 0.7\frac{MR}{(K + k_1)\, k_2}$$

Steady state error

$$s_{\text{goal}} - s_\infty = \frac{k_1}{K + k_1} s_{\text{goal}}$$

- k_1 is determined by the motor physics: $e = k_1 s$
- Large K brings the state close to desired, but we cannot put an infinite voltage into the motor!
- For any sensible K the system will undershoot the target velocity.





Proportional (P) Control

- Convenient, simple, powerful (fast and proportional reaction to errors)
- No need for modelling (sign of K must be known and the order of magnitude)
- Problem: Steady state error
- Other problems: May lead to oscillations about the goal state (later!)
- From now on K will be called K_p

Proportional Integral (PI) Control

If we could estimate this error we could add it to the control signal:

$$V_B = K_p \left(s_{\text{goal}} - s \right) + \varepsilon$$

The best way to estimate it is to integrate the error over time:

$$\varepsilon = \int \left(s_{\text{goal}} - s \right) dt$$

Obtain new control law:

$$V_B = K_p \left(s_{\text{goal}} - s \right) + K_i \int \left(s_{\text{goal}} - s \right) dt$$

Basically, this sums some fraction of the error until the error is reduced to zero.

With careful choice of K_p and K_i this can eliminate the steady state error.

Back to PID control: Steady state error – Load droop

- If the system has to hold a load against gravity, it requires a constant torque
- Similar to steady state error in the first order system
- But P controller cannot do this without error, as torque is proportional to error (so, needs some error)
- If we knew the load L could use

$$T = L - K_p\theta$$

- But in practice this is not often possible

Proportional integral (PI) control

As before, we integrate the error over time, i.e.

$$L = K_i \int_0^t \left(\theta_{\text{goal}} - \theta\left(t'\right)\right) dt'$$

Effectively this gradually increases L until it produces enough torque to compensate for the load

PI copes with load droop

Towards PID control

- Proportional control reduces large errors: Fast and powerful
- Proportional integral: Precise and delicate, deals with remaining errors if they accumulate
- What else might happen?
  - Large errors may accumulate as well such that PI can overshoot
  - The state can oscillate about the the goal state

Outlook:

- Solution of the oscillation (ringing) problem: Dampen oscillations by "artificial friction" which will be provided by derivative control $\rightarrow$ PD
- We will have three modes of feedback control: P, PI, PD and finally arrive at PID control is the combination of P, PI, PD

References

For the research on robot juggling see:

- Rizzi, A.A. & Koditschek, D.E. (1994) Further progress in robot juggling: Solvable mirror laws. *IEEE International Conference on Robotics and Automation*, p. 2935-2940
- S. Schaal and C.G. Atkeson. Open loop stable control strategies for robot juggling. In *Proc. IEEE Conf. Robotics and Automation*, pages 913 –918, Atlanta, Georgia, 1993.

IVR: Feedback control
Towards PID control

M. Herrmann

informatics

Overview

- Overshoot and oscillations in control systems
- Second order differential equations as models for systems with oscillations
- Proportional Derivative control to complement Proportional error control and Proportional Integral control towards PID control

Towards PID control

- Proportional control reduces large errors
- Proportional integral control deals with remaining errors if they accumulate
- What else might happen?
  - Large errors may accumulate as well such that PI can overshoot
  - The state can oscillate about the the goal state
    - due to inertia
    - due to delays

Oscillations in Feedback system due to time delays

How do Oscillations come about?

Imagine trying to move a robot to some zero position with the simple control law:

- If $x_t < -\delta$ meters, move forward at 1 m/sec
- If $x_t > \delta$ meters, move backward at 1 m/sec
- If $-\delta < x_t < \delta$, then stop

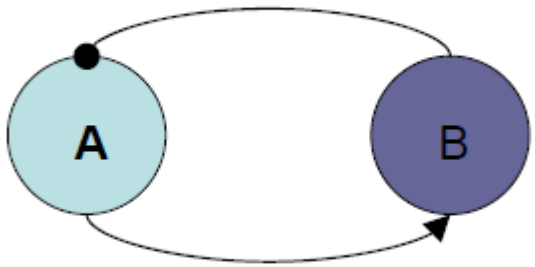
What happens if there is a delay in feedback that exceeds 2δ seconds?

Oscillations are not all bad: Central Pattern Generators

Many movements in animals, and robots, are rhythmic, e.g. walking

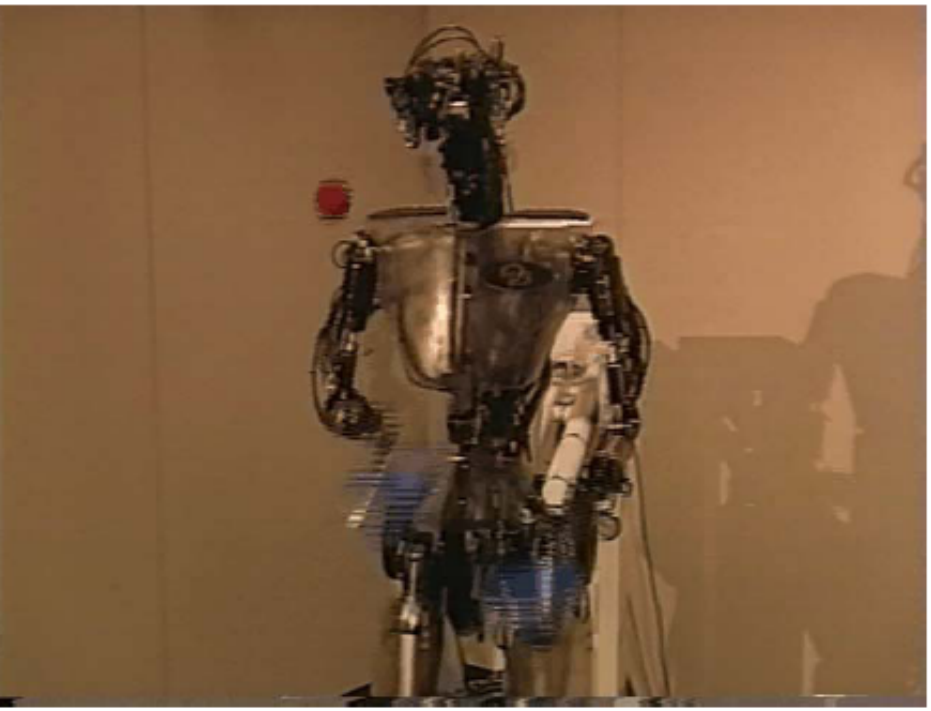
Rather than explicitly controlling position, can exploit an oscillatory process, e.g.

If A is tonically active, it will excite B
When B becomes active it inhibits A
When A is inhibited, it stops exciting B
When B is inactive it stops inhibiting A



Useful not only for control of periodic repeated movements but possibly also as elements for complex movement sequences.

CPGs in Robotics



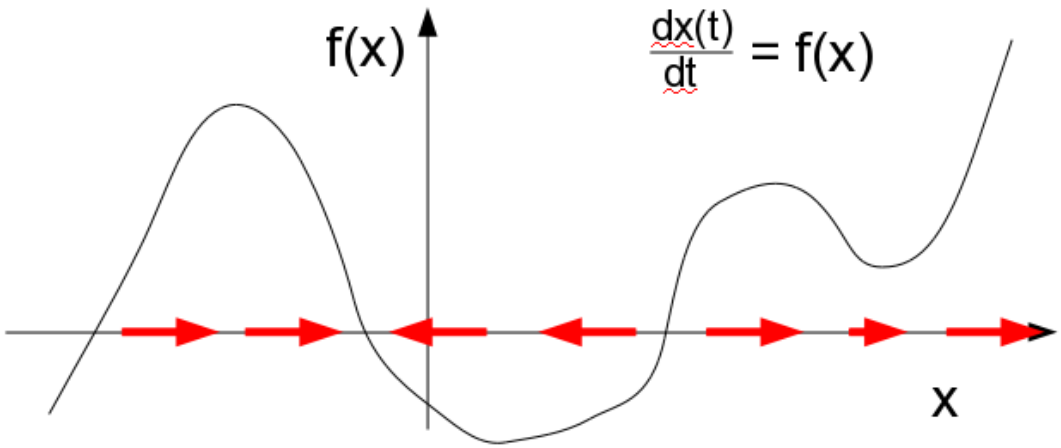
www.sarcos.com

Auke Ijspeert: Bastian

Oscillations in Feedback Systems

- In general, a time-lag in a feedback loop will result in overshoot and oscillation.
- Sometimes we want oscillation, e.g. CPG
- Depending on the dynamics, the oscillation could fade out, continue or increase.
- Note that integration introduces a time delay
- Time delays are equivalent to energy storage e.g. inertia will cause similar effects.
- We should study control systems with inertia ...

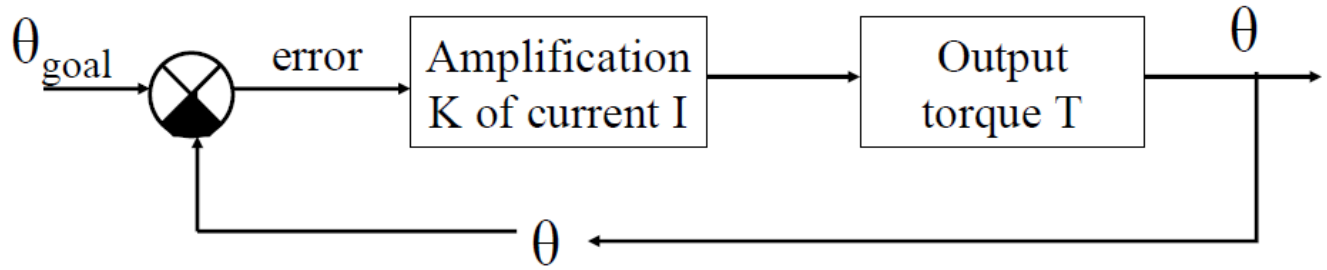
Oscillations in General Dynamical Systems



arrows indicate whether x(t) increases or decreases

- For a first order differential equation of **one variable and without delay** the present state determines uniquely the behaviour. Therefore, the system **cannot be oscillatory**.
- Second order differential equations (or systems of dim ≥ 2) can describe systems where oscillations are possible. Solutions of such differential equations are characterised by an initial velocity in addition to the initial state.

Second order system



Want to move a simple robot arm to a desired angular position θ_{goal}

For a DC motor on a robot joint

$$T = J\frac{d^2\theta}{dt^2} + F\frac{d\theta}{dt}$$

where J is the inertia of the joint and F is the joint friction

Proportional control

$$K_p\left(\theta_{\text{goal}} - \theta\right) = T$$

Second order system solution

For simplicity let $\theta_{\text{goal}} = 0$. Then the system process is

$$K_p\theta = J\frac{d^2\theta}{dt^2} + F\frac{d\theta}{dt}$$

The solution to this equation has the form (insert and check)

$$\theta = e^{-\frac{F}{2J}t}\left(c_1 e^{\frac{\omega}{2}t} + c_2 e^{-\frac{\omega}{2}t}\right)$$

where

$$\omega = \sqrt{\frac{F^2}{J^2} - \frac{4K_p}{J}}$$

which may be complex $\Longrightarrow$ oscillatory solution

c_1, c_2 can be determined from the initial state and initial velocity.

A remark on linear differential equations

Second-order system

$$\frac{d^2\theta}{dt^2} = A_1 \frac{d\theta}{dt} + A_0\theta + C$$

Define $\rho = \frac{d\theta}{dt}$ and insert

$$\frac{d\rho}{dt} = A_1 \frac{d\theta}{dt} + A_0\theta + C$$

We get an equivalent two-dimensional first-order system

$$\begin{pmatrix} \frac{d\rho}{dt} \\ \frac{d\theta}{dt} \end{pmatrix} = \begin{pmatrix} A_1 & A_0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \rho \\ \theta \end{pmatrix} + \begin{pmatrix} C \\ 0 \end{pmatrix}$$

Diagonalize:

$$R\begin{pmatrix} \frac{d\rho}{dt} \\ \frac{d\theta}{dt} \end{pmatrix} = R\begin{pmatrix} A_0 & A_1 \\ 1 & 0 \end{pmatrix} R^{-1} R\begin{pmatrix} \rho \\ \theta \end{pmatrix} + R\begin{pmatrix} C \\ 0 \end{pmatrix} \text{ such that } R\begin{pmatrix} A_0 & A_1 \\ 1 & 0 \end{pmatrix} R^{-1} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$$

Transform to the new coordinates, solve two simple diff-eqs with constant terms, transform back to ρ,θ, insert initial conditions, done

Over- and Under-Damping

The system behaviour depends on J, F, K_p as follows:

$$\theta = e^{-\frac{F}{2J}t}\left(c_1 e^{\frac{\omega}{2}t} + c_2 e^{-\frac{\omega}{2}t}\right) \quad \text{with} \quad \omega = \sqrt{\frac{F^2}{J^2} - \frac{4K_p}{J}}$$

- The system returns to the goal with under-damped, sinusoidal behaviour for
$$\frac{F^2}{4K_p} < J$$
- The system returns to the goal with over-damping
$$\frac{F^2}{4K_p} > J$$
- The system returns to the goal (critical damping)
$$\frac{F^2}{4K_p} = J \quad \Rightarrow \quad \text{solution becomes: } \theta = c\, e^{-\frac{F}{2J}t}$$

Proportional derivative (PD) control

Often, e.g. for large robots, inertia is large and friction small.

Consequently the system overshoots, reverses the error and control signal, overshoots again ...

To actively brake the motion, we want to apply negative torque when error is small and velocity high

Make

$$T = K_p \left(\theta_{\text{goal}} - \theta\right) + K_d \frac{d\theta}{dt} = J \frac{d^2\theta}{dt^2} + F \frac{d\theta}{dt}$$

This can be seen as P (or PI) control with artificial friction

$$K_p \left(\theta_{\text{goal}} - \theta\right) = J \frac{d^2\theta}{dt^2} + \left(F - K_d\right) \frac{d\theta}{dt}$$

Proportional derivative (PD) control

Example

$$T = J\frac{d^2\theta}{dt^2} + F\frac{d\theta}{dt}$$

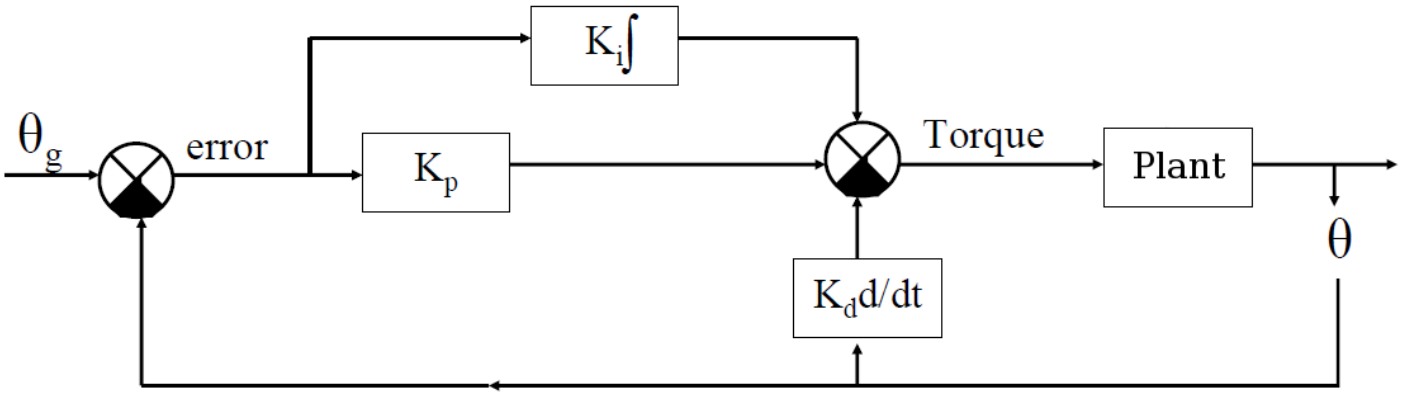
Control law (P and PD):

$$T = K_p\left(\theta_{\text{goal}} - \theta\right) + K_d\frac{d\theta}{dt}$$

- Dampens oscillations, improves stability
- Useful for large inertia & small friction (adds artificial friction)
- Derivative term should be $\frac{d}{dt}\left(\theta_{\text{goal}} - \theta\right)$, use $-K_d$ in this case
  Derivative of θ can improve stability even when θ_{goal} changes
- Possible problems:
  - Derivative is sensitive to measurement noise
  - Tends to slow down the control action
  - D-term does not contain information about the (constant) goal state $\rightarrow$ usually in combination with other control signals

Combine as PID control (Three mode control)

Black sectors denote negation



$$T = K_p\left(\theta_{\text{goal}} - \theta\right) + K_i\int_0^t \left(\theta_{\text{goal}} - \theta\left(t'\right)\right)dt' + K_d\frac{d\theta}{dt}$$

- How do K_p, K_i, K_d interact?
- How to choose K_p, K_i, K_d?
- PID controllers are used in more than 95% of continuous feedback control applications (as of 1995)

Summary

- Proportional error control: Fast and powerful
- Proportional Integral control: Precise and delicate
- Proportional Derivative control: Stabilising (unless the system is rather noisy)
- Usually combinations, i.e. PD, PI or PID, are used

IVR: PID control

M. Herrmann

informatics

Overview

- Proportional error + Proportional Integral + Proportional Derivative = PID
- Effects of the gain factors
- Non-stationary targets
- PID tuning

Combining controllers

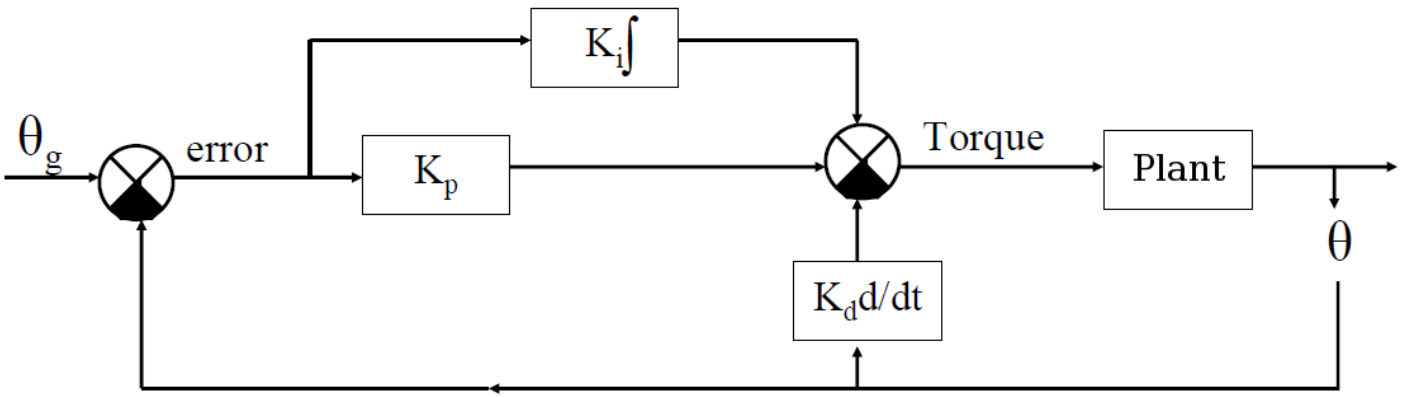
Control signals can chosen:

- Proportional to error (P)
- Proportional to error Integral (I)
- Proportional to error Derivative (D)

How do the different modes of the set-point interact?

Combine as PID control (Three mode control)

Black sectors denote negation



$$T = K_p\left(\theta_{\text{goal}} - \theta\right) + K_i\int_0^t \left(\theta_{\text{goal}} - \theta\left(t'\right)\right)dt' + K_d\frac{d\theta}{dt}$$

- PID controllers are used in by far the most continuous feedback control applications
- How to choose K_p, K_i, K_d?

Characterising the behaviour of a control system (SASO)

1. Stability: Returns to set point after (small) perturbations
2. Accuracy (Steady-state error): the difference between the steady-state output and the desired output.
3. Settling time: time it takes for the system to converge to its steady state
   Rise time: time it takes for the plant output to rise beyond 90% of the desired level for the first time
   - May be long in the case of on-going oscillations
   - Rise time and settling time replace half time (which was meaningful only for non-oscillatory exponential convergence)
4. Overshoot: how much the the peak level is higher than the steady state, normalised against the steady state.

Gingham Zhong: PID Controller Tuning: A Short Tutorial
saba.kntu.ac.ir/eecd/pcl/download/PIDtutorial.pdf

PID control example

Second-order system

$$T(t) = J\frac{d^2\theta}{dt^2} + F\frac{d\theta}{dt}$$

Controller

$$K_p\left(\theta_{\text{goal}} - \theta(t)\right) + K_i\int\left(\theta_{\text{goal}} - \theta(t)\right)dt + K_d\frac{d\theta}{dt} = T(t)$$

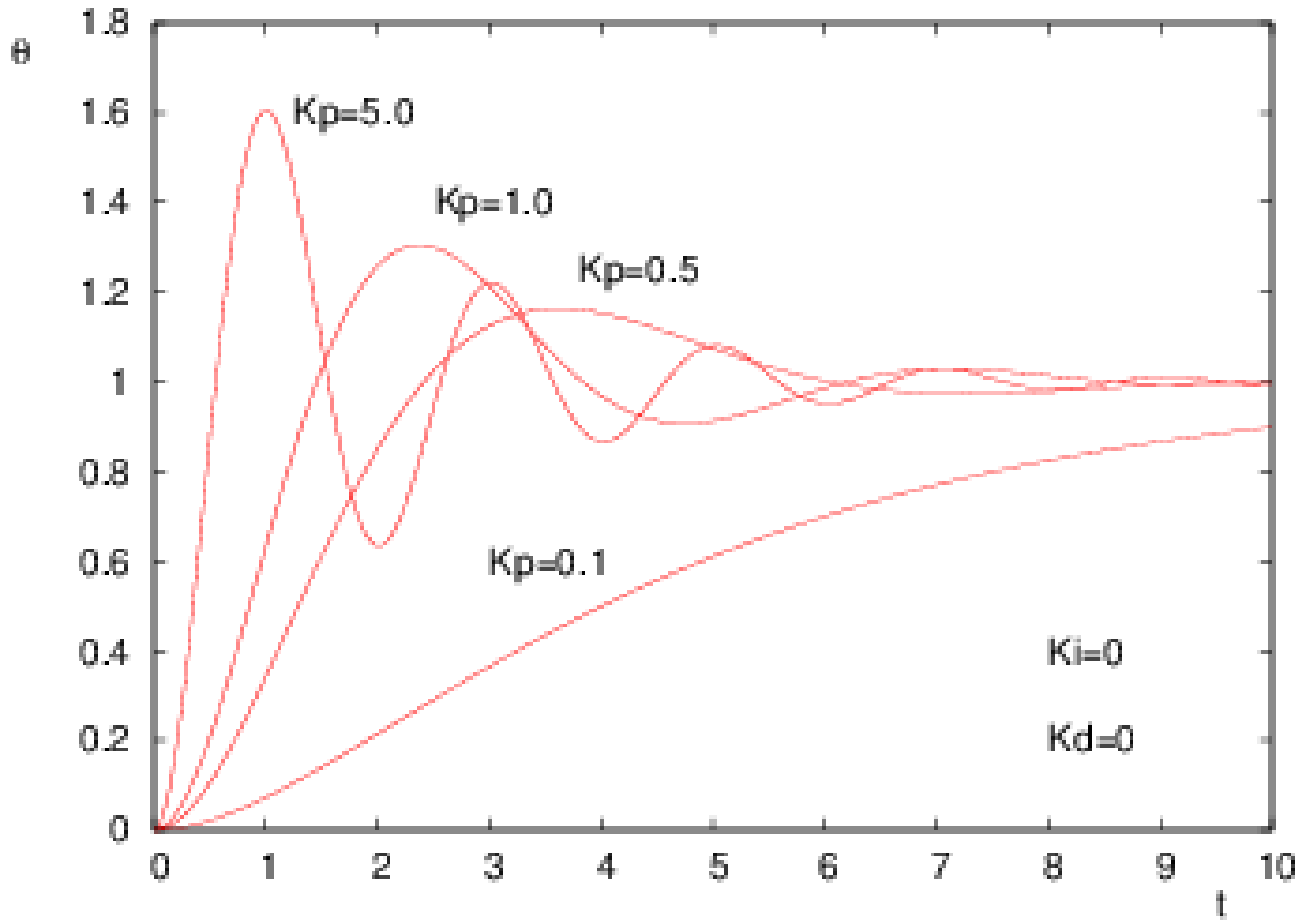
Numerical solution with:

$A_1 = \frac{K_d - F}{J}$, $A_0 = -\frac{K_p}{J}$, $C = \frac{K_i}{J}\int_0^t\left(\theta_{\text{goal}} - \theta(t)\right)dt + \frac{K_p}{J}\theta_{\text{goal}}$

$$\frac{d^2\theta}{dt^2} = A_1\frac{d\theta}{dt} + A_0\theta + C$$

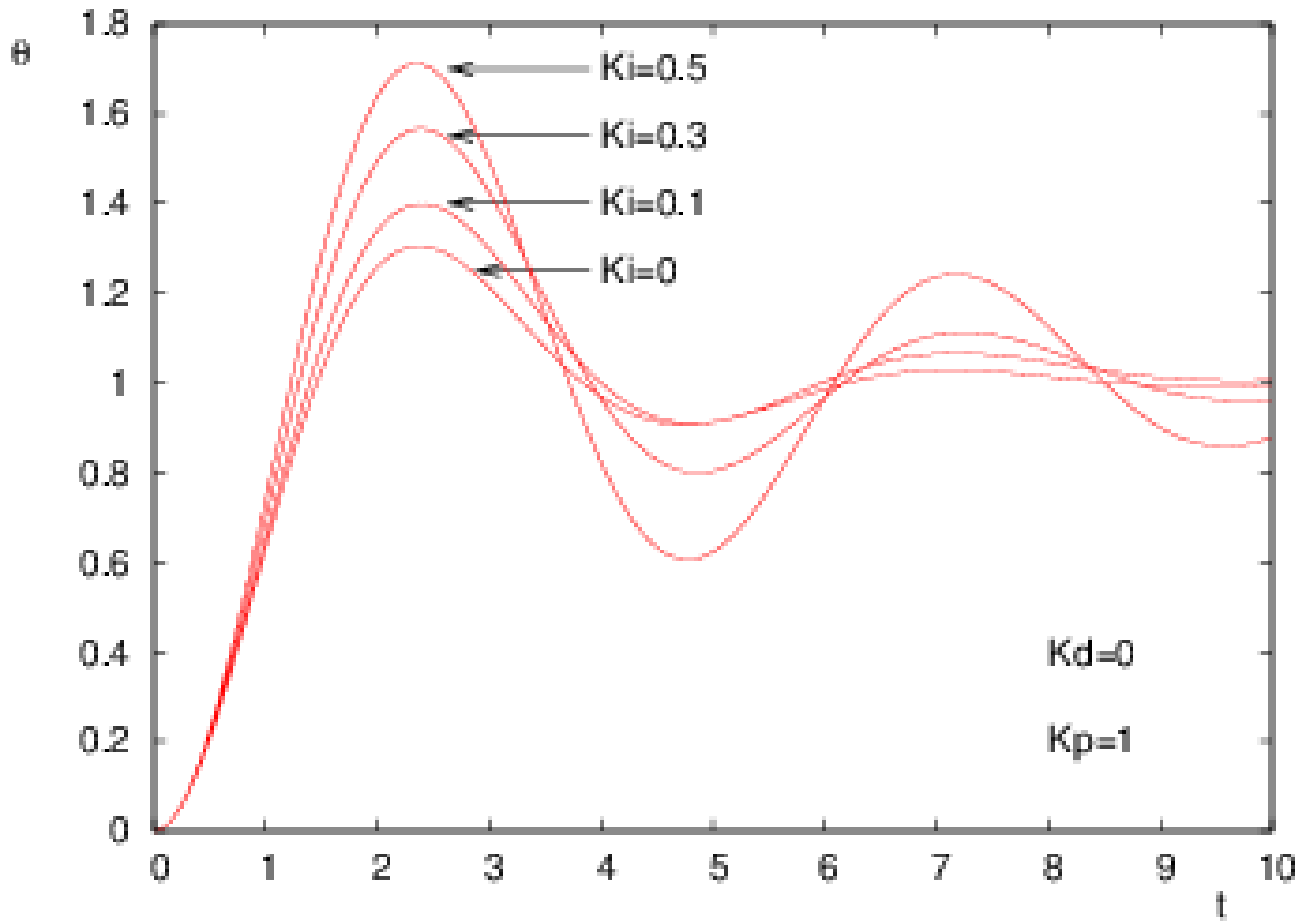
PID control example

1. Behaviour at $K_p = K_i = K_d = 0$: Choose J and F such that the system is overdamped (see 2nd order diff-eq. example)
2. Start with proportional control: $K_i = K_d = 0$, $\theta_{\text{goal}} = 1$



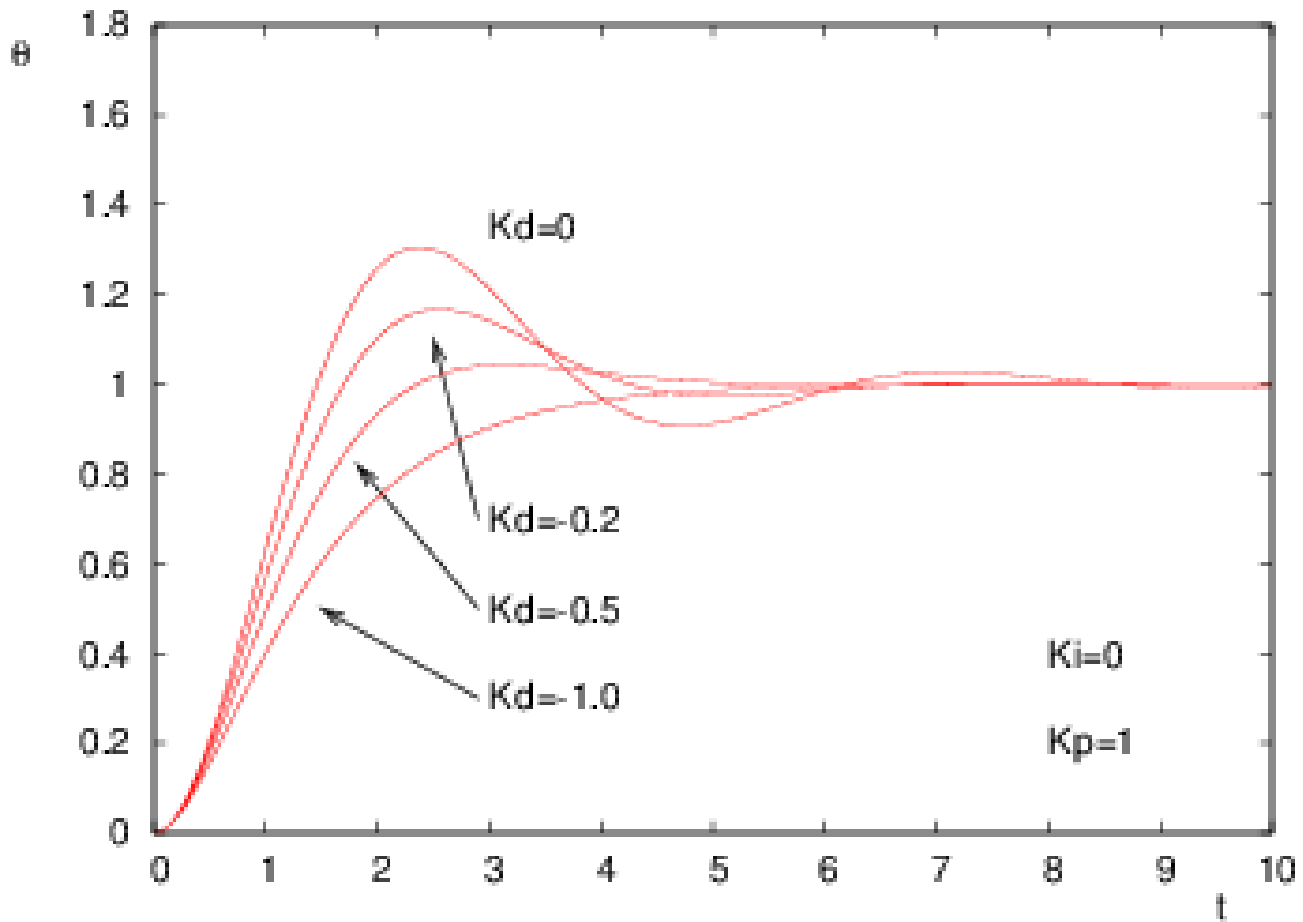
PID control example

3. Choose a reasonable rise time, e.g. $K_p = 1$, now vary K_i: Similar effect as increasing K_p



PID control example

4. K_i did not help, so set $K_i = 0$ and $K_p = 1$, and vary K_d: Reduces oscillations, rise time still low (e.g. at $K_d = 0.5$)



Note, that the negative sign of K_d is for consistency with the equation two slides back. Considered as artificial friction, the used differential term is clearly damping.

PID control example: Evaluation

- Often PD (proportional + derivative) control is sufficient
- Integral term needed only if steady state error is expected or if the system is noisy
- Consider priorities when determining overshoot:
  - When catching a ball: fast rise time is essential (could set goal state to a lower value and make sure that the ball arrives at the overshoot)
  - When moving towards a position near an obstacle: slow rise time, overdamped movement $\rightarrow$ no overshoot
  - In many other cases: Adjust K_d to realise critical damping

Effects of increasing a parameter independently

Parameter	Rise time	Overshoot	Settling time	Steady-state error	Stability
K_p	Decrease	Increase	Small change	Decrease	Degrade
K_i	Decrease	Increase	Increase	Decrease significantly	Degrade
K_d	Minor increase	Decrease	Minor change	No effect in theory	Improve (if K_d is small)

http://en.wikipedia.org/wiki/PID_controller (except red entries)

Typical steps for designing a PID controller

- Determine what characteristics of the controlled system need to be improved
- Use K_p to decrease the rise time.
- Use K_i to eliminate the steady-state error.
- Use K_d to reduce the overshoot and settling time.

This works in many cases

JFYI: Ziegler-Nichols tuning rule (reaction curve method)

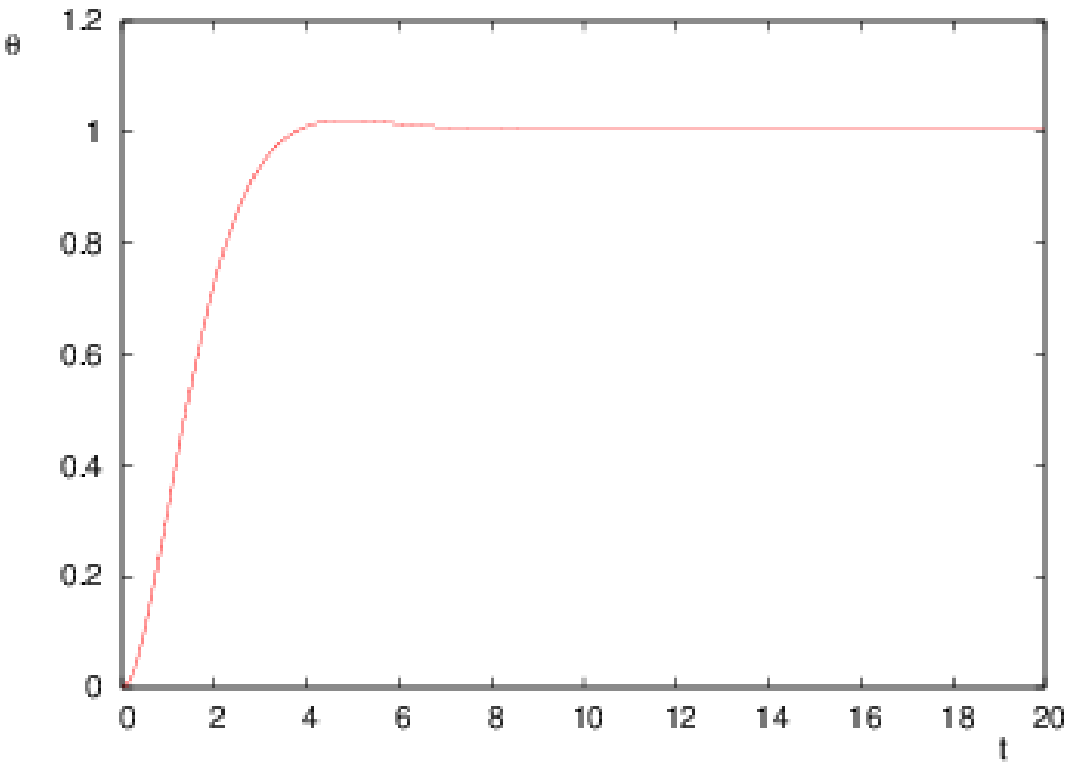
- Practical control method, i.e. the controlled system is accessed experimentally

1. Set I and D gains to zero.
2. Check sign of gain (say positive)
3. Increase P gain (from zero) until until output starts to oscillate $\rightarrow$'ultimate gain' K_u and oscillation period T_u
4. Use K_u and T_u to set K_p, K_i and K_d based on heuristic values: $K_p = 0.65K_u$, $K_i = 2K_p/T_u$ and $K_d = K_pT_u/8$

- May create some overshoot
- Stable to disturbances
- Not very good in tracking tasks
- Not equally good in all applications

Ziegler-Nichols rule tested

- Second order system ($F = 0.5$, $J = 0.5$)
- $K_u \approx 1$: one full oscillation period visible $\Rightarrow$ $T_u = 5$
- $\Rightarrow$ $K_p = 0.65$, $K_d = 1.25$, $K_i = 0.26$ (scaled by time step!)



- Performance similar as tuned by hand (Z-N rule works best for first-order systems!)

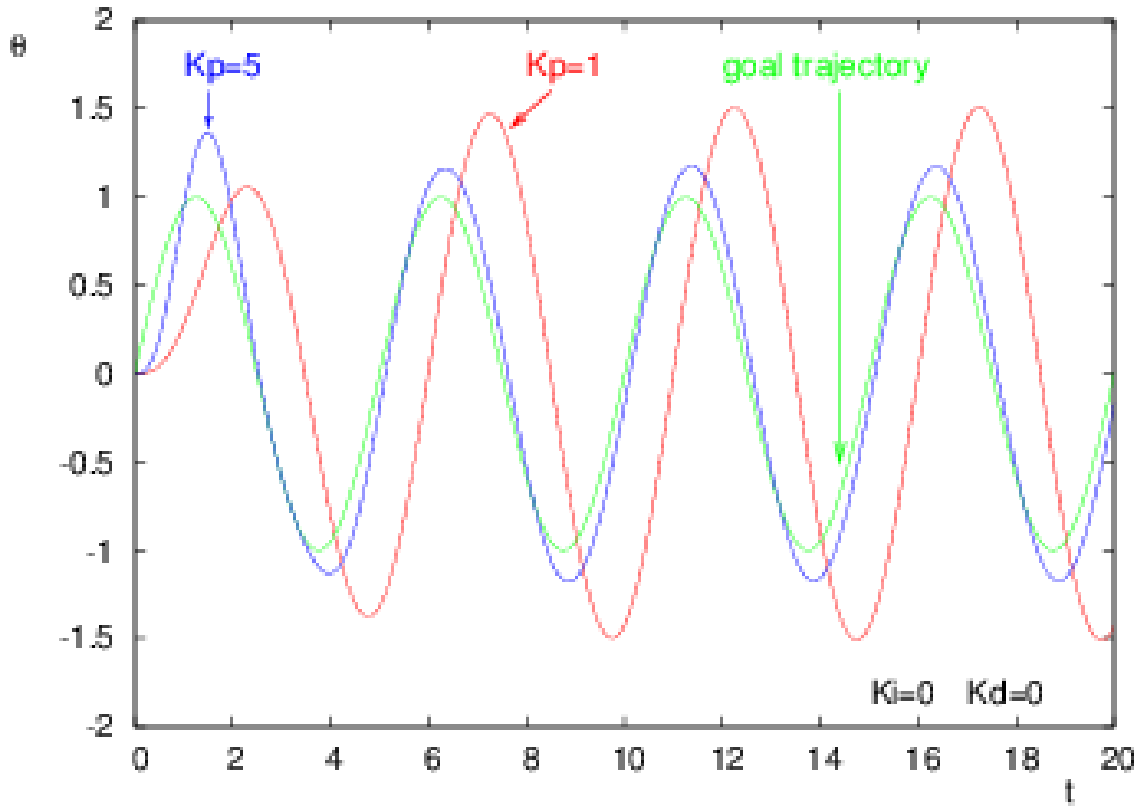
Why Do We Keep Hinting That Results are Lousy? (http://www.mstarlabs.com/control/znrule.html)

Limitations of PID control

- PID control is usually the best controller with no model of the process (PID can be used on top of model-based control)
- Does not provide optimal control
- Is only reactive, may be slow, and needs errors to be able to react (combine with feed-forward control or forward models)
- D-term may suffer from intrinsic or measurement noise ($\Rightarrow$ use low-pass filter)
- D-term (error derivative) and I-term may suffer from sudden set-point changes ($\Rightarrow$ use set-point ramping)
- Is tuned to a particular working regime ($\Rightarrow$ gain scheduling)
- Is linear and (anti)symmetric (e.g. usually a heating system does not involve symmetric cooling)

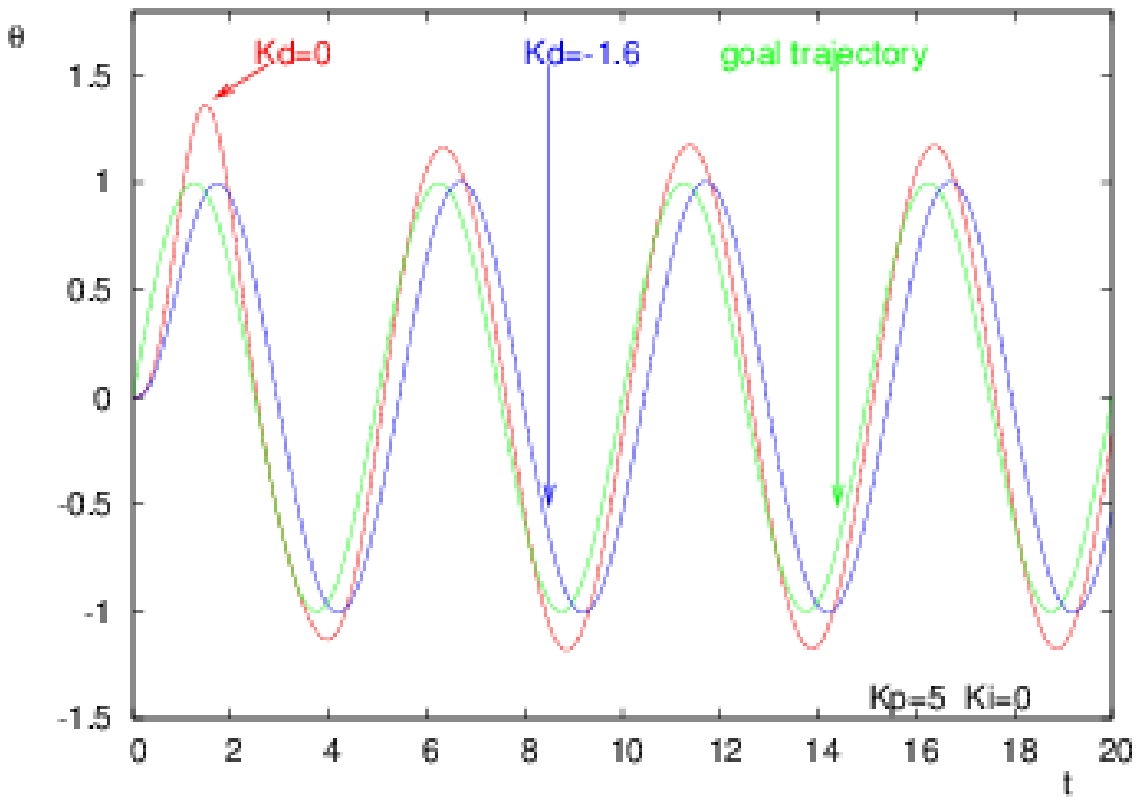
Control with changing set points

- In tracking tasks the set point changes continuously: goal trajectory
- Rise time and settling time appear as delays (phase shifts) which depend on the rate of change of the goal trajectory
- Delays can be reduced by high-gain proportional control



Control with changing set points

- Differential feedback can reduce the overshoot, but tends to increases phase shift
- Integral feedback will not improve the situation
- Solution: Forward models



Changing set-points: Possible improvements

- Set-point switching: Change the set point in a ramp-like way
- Initialising the integral term at a suitable value
- Disabling the integral function until the state has entered the controllable region
- Limiting the time period over which the integral error is calculated
- Preventing the integral term from accumulating above or below pre-determined bounds
- For a constant set-point, the D-term can be either $\frac{d}{dt}\theta$ or $\frac{d}{dt}\left(\theta_{\text{goal}} - \theta(t)\right)$, now the second form should be considered

http://en.wikipedia.org/wiki/PID_controller

More on control (JFYI)

- Non-linear control: How to deal with complex systems?
  - Linearisation, gain-scheduling, Lyapunov stability, sliding-mode control. ...
- Robust control: Uncertain parameters or disturbances?
  - H_∞ control: Stabilisation (and other desired properties) with guaranteed performance, ...
- Adaptive control: Changing environments?
  - System identification, model-based control, self-tuning, ...
- Distributed control: Communication, negotiation and all of the above
- Intelligent control: How to deal with all the remaining cases?
  - Use AI, learning, evolutionary algorithms etc.

Further reading on control theory and applications

Most standard control textbooks discuss PID control, e.g.: Andrew D. Lewis: A Mathematical Approach to Classical Control. 2003. www.mast.queensu.ca/~andrew/teaching/math332/notes.shtml

igor.chudov.com/manuals/Servo-Tuning/PID-without-a-PhD.pdf

IVR: A Brief Outlook to Robot Architectures

M. Herrmann

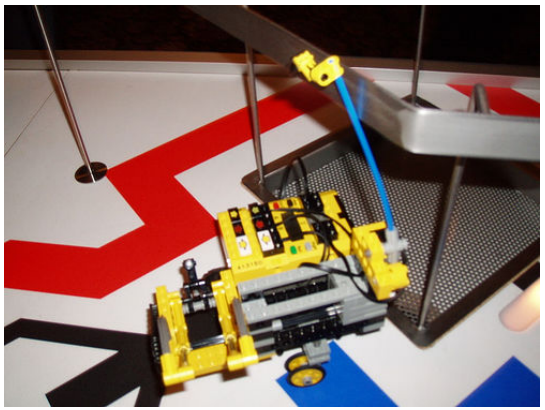
informatics

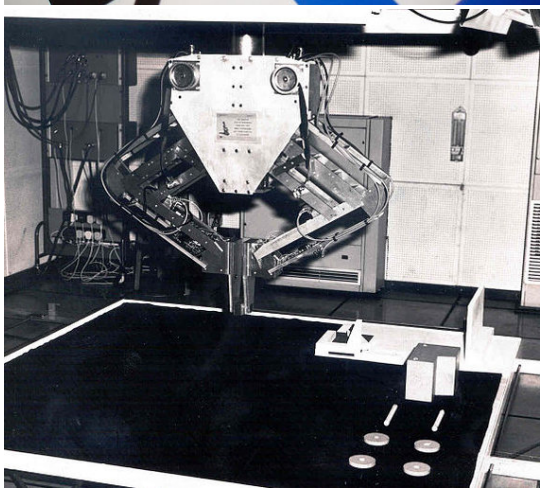
Overview

- The Sense-Plan-Act paradigm
- Brooks' subsumption architecture
- Three-layer architecture

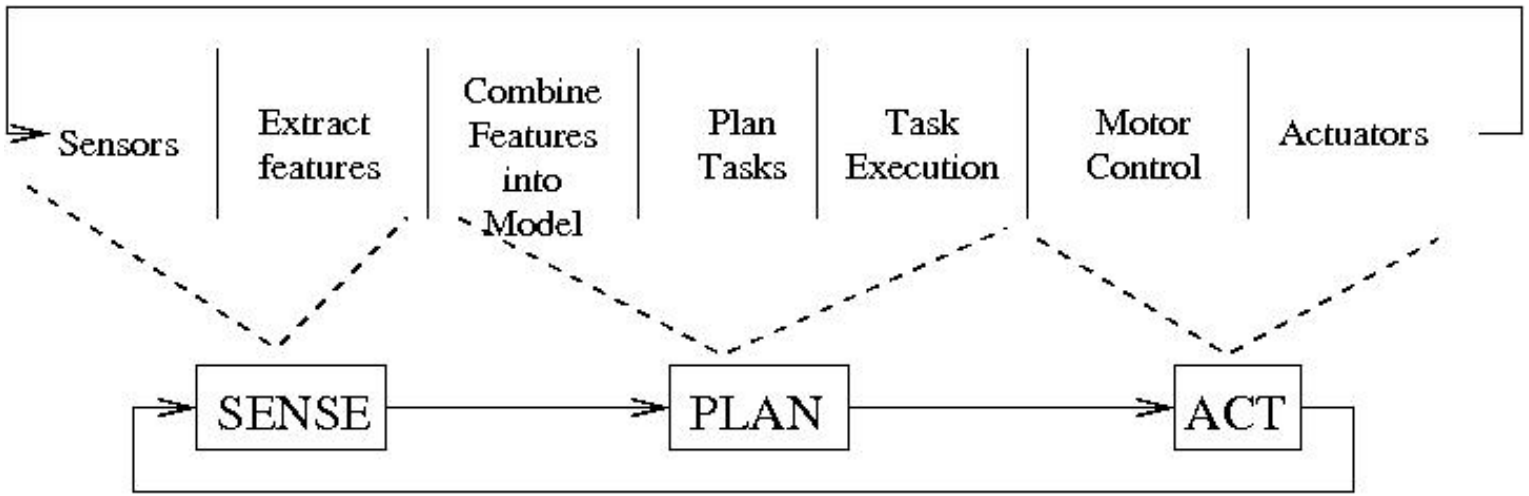
Behaviour-based robotics

- Reactive behaviours
- Embodied intelligence
- Action-relevant sensory information processing
- Minimal processing, representation, control
- Modular, hierarchical, distributed control





Beyond the "SPA" Control Paradigm



SPA is

- serial
- ad hoc
- analytical
- assumptious

SPA lacks:

- speed and efficiency
- flexibility and adaptivity
- modularity and scalability
- error–tolerance and robustness

Reactive control

- Direct coupling of perception and action. Aim is timely robotic response in dynamic and unstructured worlds. Typically in the context of motor behaviours.
- Reactive control architectures will be discussed on more detail with the behaviour based control architectures.
- One of the prominent examples for reactive control is the subsumption architecture (see below).

Rodney Brooks: "Fast, Cheap & Out of Control" (1997)

Some considerations that (according to Brooks) can be helpful:

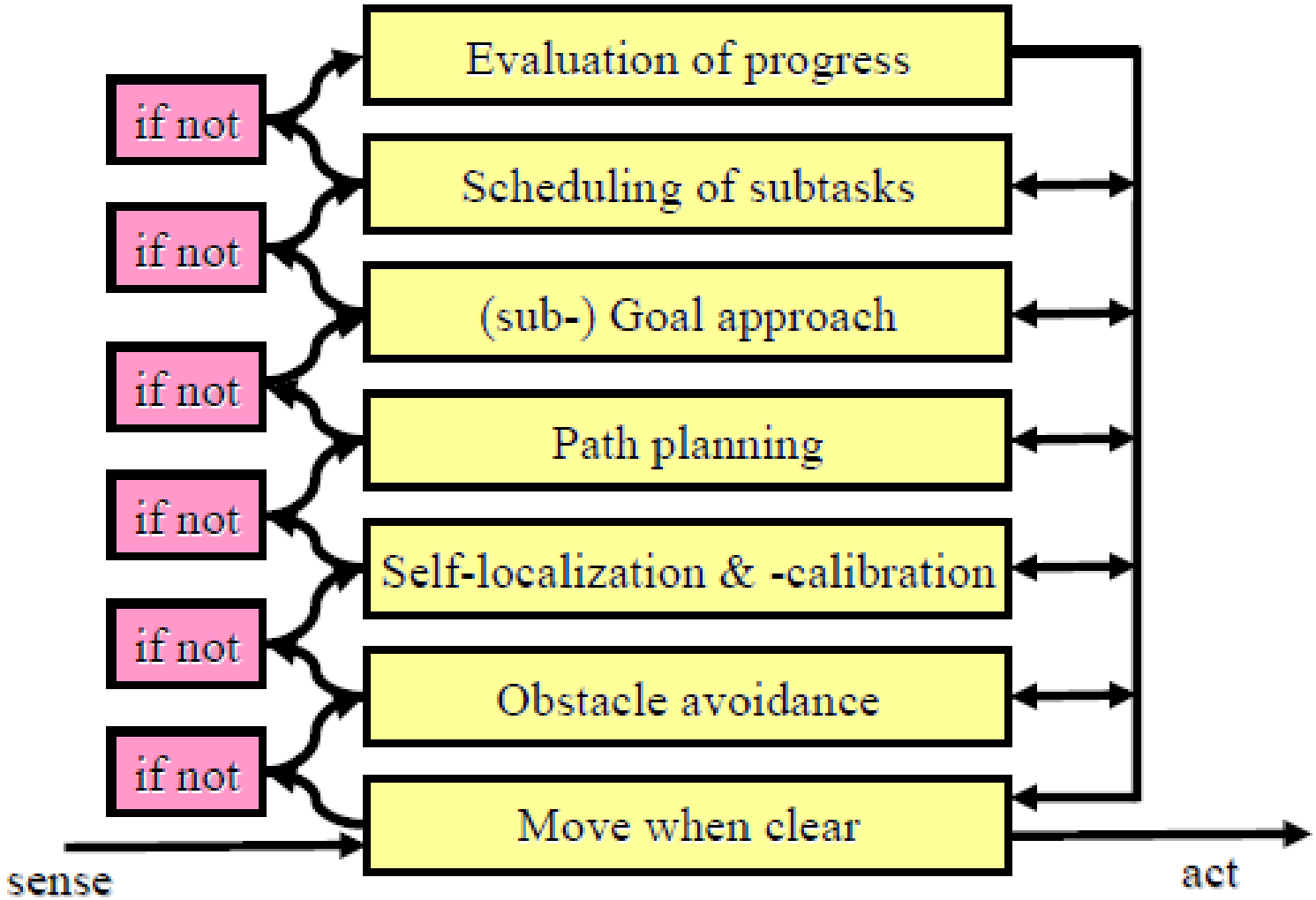
- Planning is just a way of avoiding figuring out what to do next.
- The world is its own best model
- Complex behaviour need not necessarily be the product of a complex control system
- Simplicity is a virtue
- Robots should be cheap
- All on-board computation is important
- Systems should be build incrementally
- Intelligence is in the eye of the observer
- No representation, no calibration, no complex computers

Some objections that can be misleading:

- In a different environment the robot will fail!
  "A function that deals with this problem will create more problems"
- The system cannot be debugged!
  "Bugs, too, are in the eye of the observer"
- It's not scalable!
  "Elephants don't play chess"

http://people.csail.mit.edu/brooks/index.html

Subsumption Architecture (1986)



Evaluation of the Subsumption Architecture

- "I wouldn't want one to be my chauffeur" (C. Torpe)
- Modifications at low-levels affect higher levels
- Often there the hierarchy is not strict and conflict may arise
- Priorities rather than inhibition
- Representations, plans, and models do help
- Reproducibility is a virtue
- SPA is top-down, Subsumption Architecture is bottom-up
- "neats vs. scruffies"

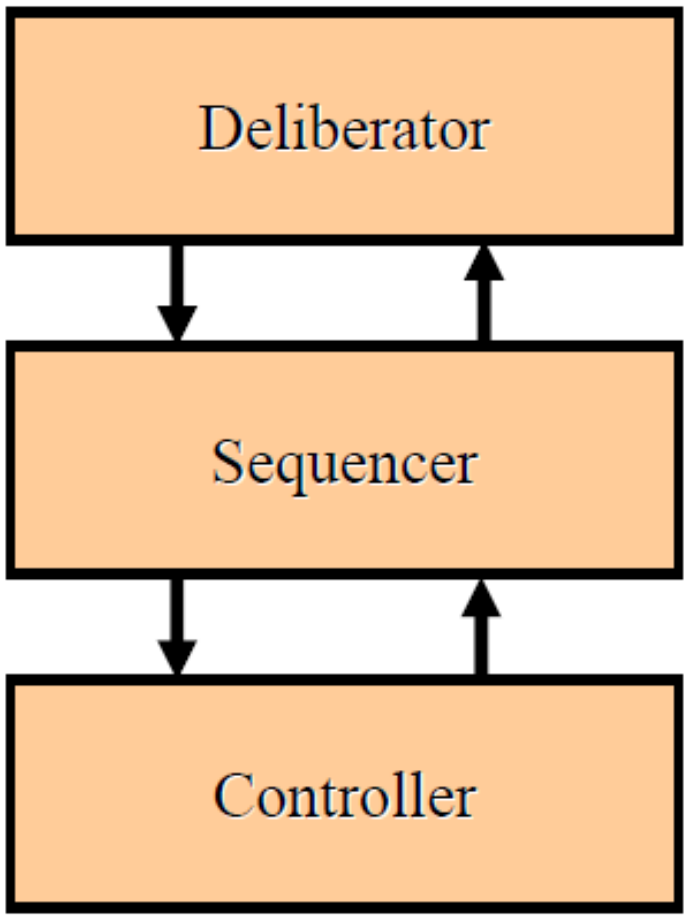
The three major control paradigms

1. Hierarchical: Sense $\rightarrow$ Plan $\rightarrow$ Act
2. Reactive: Sense $\leftrightarrow$ Act
   - Usually no planning
   - Generalises to behaviour-based architectures: Several parallel reactive loops that are activated by particular inputs or goals
   - Includes subsumption architecture
3. Hybrid deliberative/reactive: reactive loops are selected, scheduled and modified according to a high-level representation that includes planning components

Dimensions (M. Arbib)

Deliberative	Reactive
symbolic	reflexive
needs internal representation	no internal representation
complex digital computation	simple analogue computation
slow responses	real-time responses
near-complete world model	embedded
artificially intelligent (cognitive)	low-level intelligence

Three-layer architecture (TLA)



Planning, search, reasoning standard programming, time consuming computations

Competition, scheduling, and adaptation of behaviours, reactive plan execution

Elementary reactive behaviours

R. James Firby 1990. Erann Gat, 1998

Modular architectures

- Schemata (M. Arbib)
- Circuit architecture: Situated automata (L. Kaelbling)
- Action selection (P. Maes), behaviour-based robotics (R. Arkin)
- Dynamical systems, metaheuristic control, neuro-control
- Cognitive architectures

Architectures summary

1. Simplicity is a virtue
2. The subsumption architecture is simple and extensible and often makes a good start
3. The ultimate goal is to interface automatic reasoning with the real world
4. Limited resources, unreliable sensors, unpredictability and complexity of the environment are problems for any approach to robot control
5. Robotics needs vision as well as contributions from many other disciplines