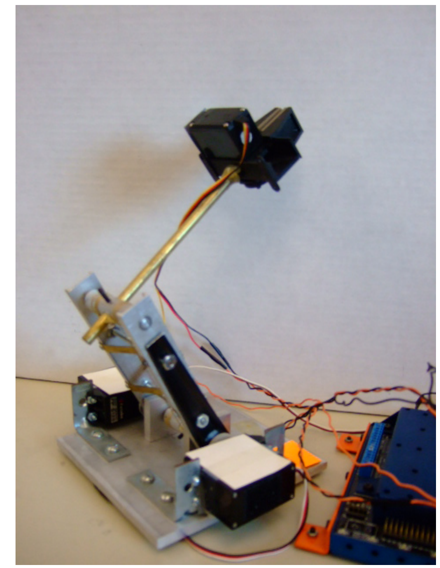
CMPUT 412 Review

WAM $100,000

Martin Jagersand

Camilo Perez

University of Alberta

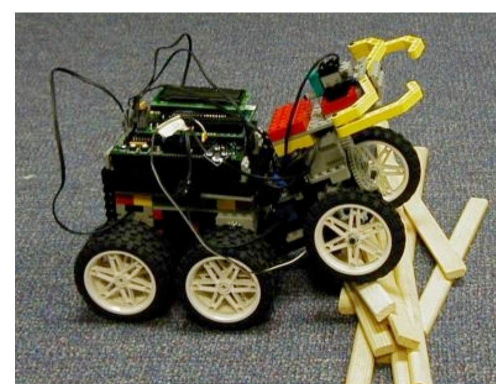


iRobot

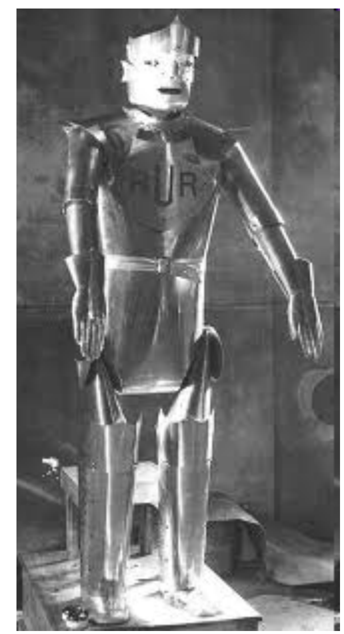
Packbot $100,000

Student robots

$100-500



Robot?



Robotics broad field:

Arms, hands, rover, UAV,…

How do we categorize these robots?

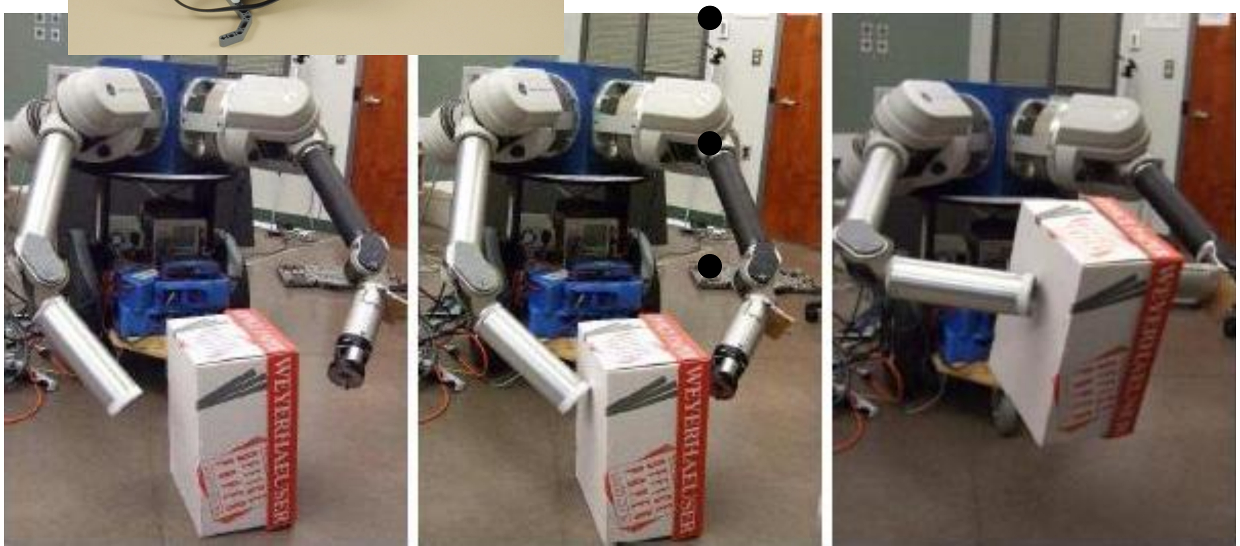
- What they can do?
  - Most robots can move things (but varying distances)
- What sensors they have?
  - We can generally equip any robot with any type of sensor.
- How they move
  - The motion properties of an arm is different from a mobile robot/car and a UAV/helicopter

Mobile Robots (ground)

- Moves on 2D ground surface
- Needs just 2 motors
- Inexpensive
- Easy to model and control
- Large range of motion
- Hard to exactly localize
- Cannot generally pick things up and manipulate them

Robot arms and hands (linkages)

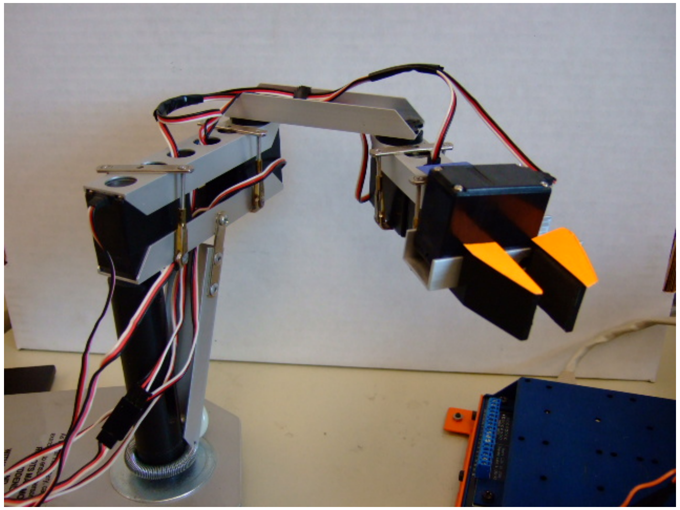
- Moves in 6DOF (3D position, 3D orientation)
- Min 6 motors in linkage
- (near) Perfect localization
- Expensive
- Easy to control
- Manipulates!

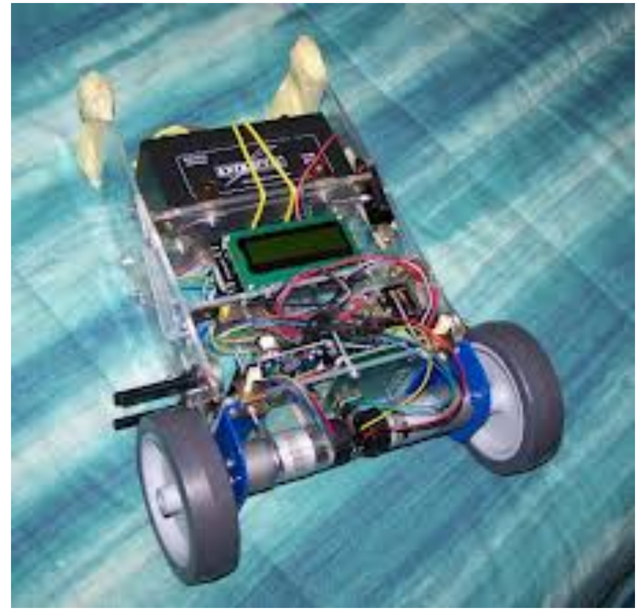


Free-flying robots
Aerial and Underwater

- Moves in 6DOF (3D position, 3D orientation)
- Generally underactuated (4DOF)
- Hard to model and control
- User error = Crash and break
- Hard to exactly localize
- RC heli inexpesive,
- Military UAV expensive

Parts of a Robot





- Motors
- Motor controllers
- Transmission
- Linkages
- Sensors
- Computer

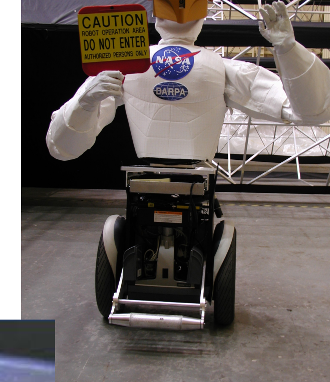
Majority Robotics in practice: Robot arms in Manufacturing

- Engineered environment
- Repeated motion

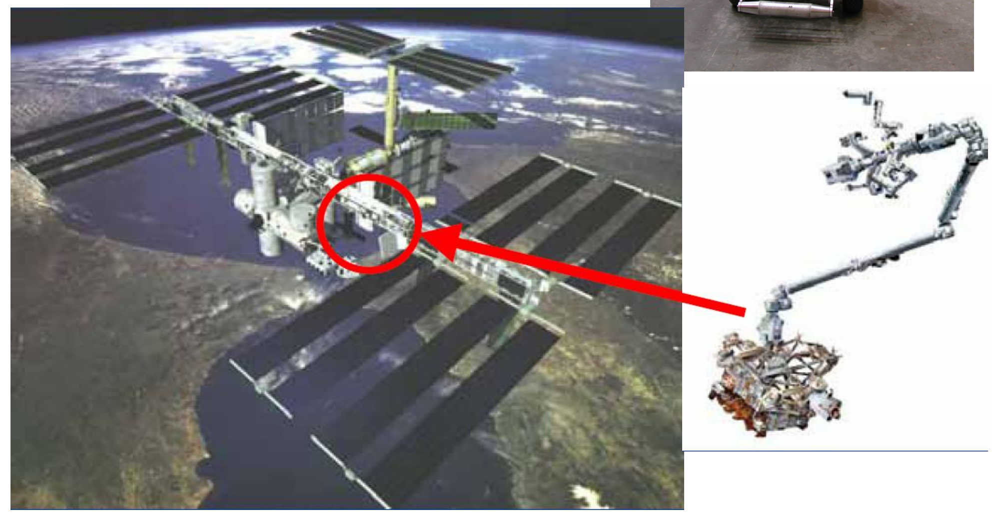
1 million arms in operation worldwide

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

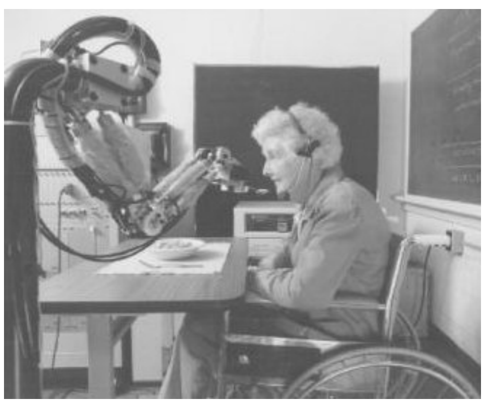
Emerging Robotics Applications



Space - in-orbit, repair and maintenance, planetary exploration anthropomorphic design facilitates collaboration with humans



Basic Science - computational models of cognitive systems, task learning, human interfaces

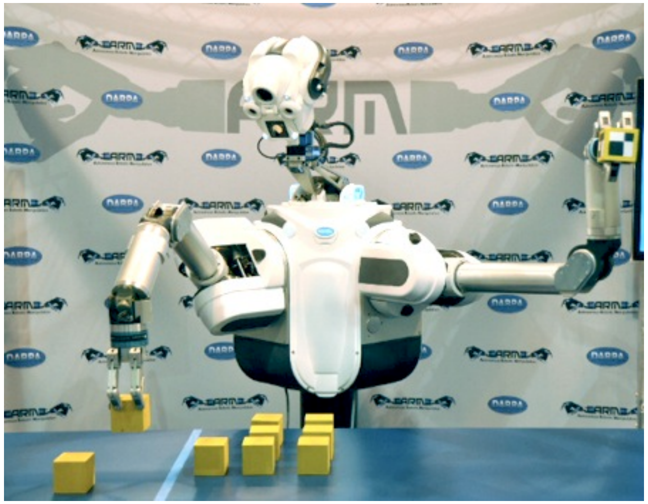


Health - clinical applications, "aging-in-place," physical and cognitive prosthetics in assisted-living facilities

Military or Hazardous - supply chain and logistics support, re-fueling, bomb disposal, toxic/radioactive cleanup

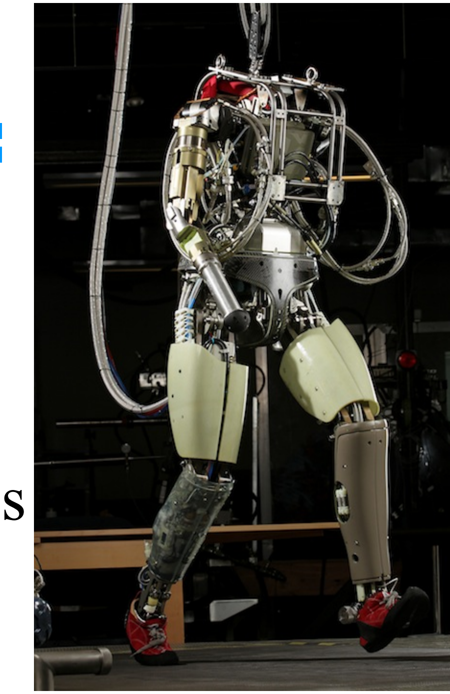
No or few robots currently operate reliably in these areas!

Newer robots



DARPA WAMs

Boston dynamics



UofA’sR obotics and Vision group

What is a robot?

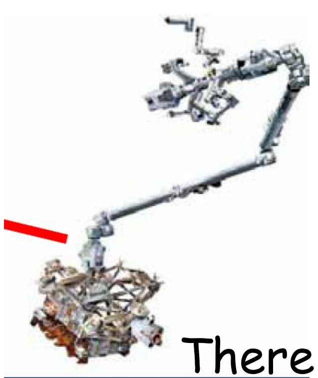
Robot : A physical system that *(semi-)* *"autonomously"* senses the environment and acts in it.

Autonomy might be a continuous, not a discrete attribute

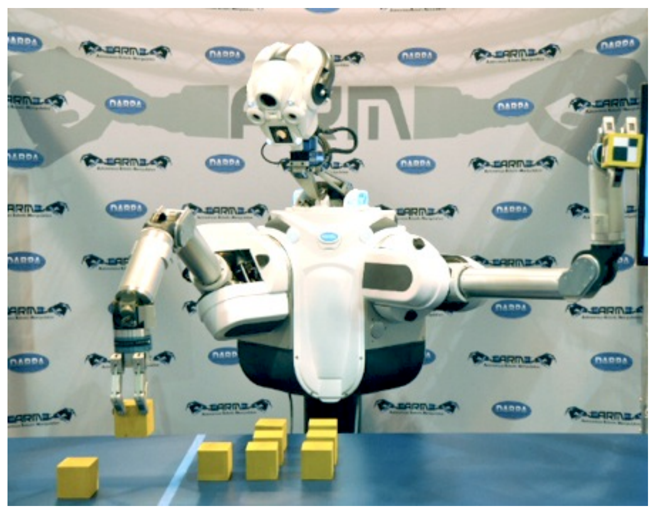
Researchers disagree on what kind and how much autonomy is needed

none → **full**

Robot Wars, Battlebots



Canada Arm2



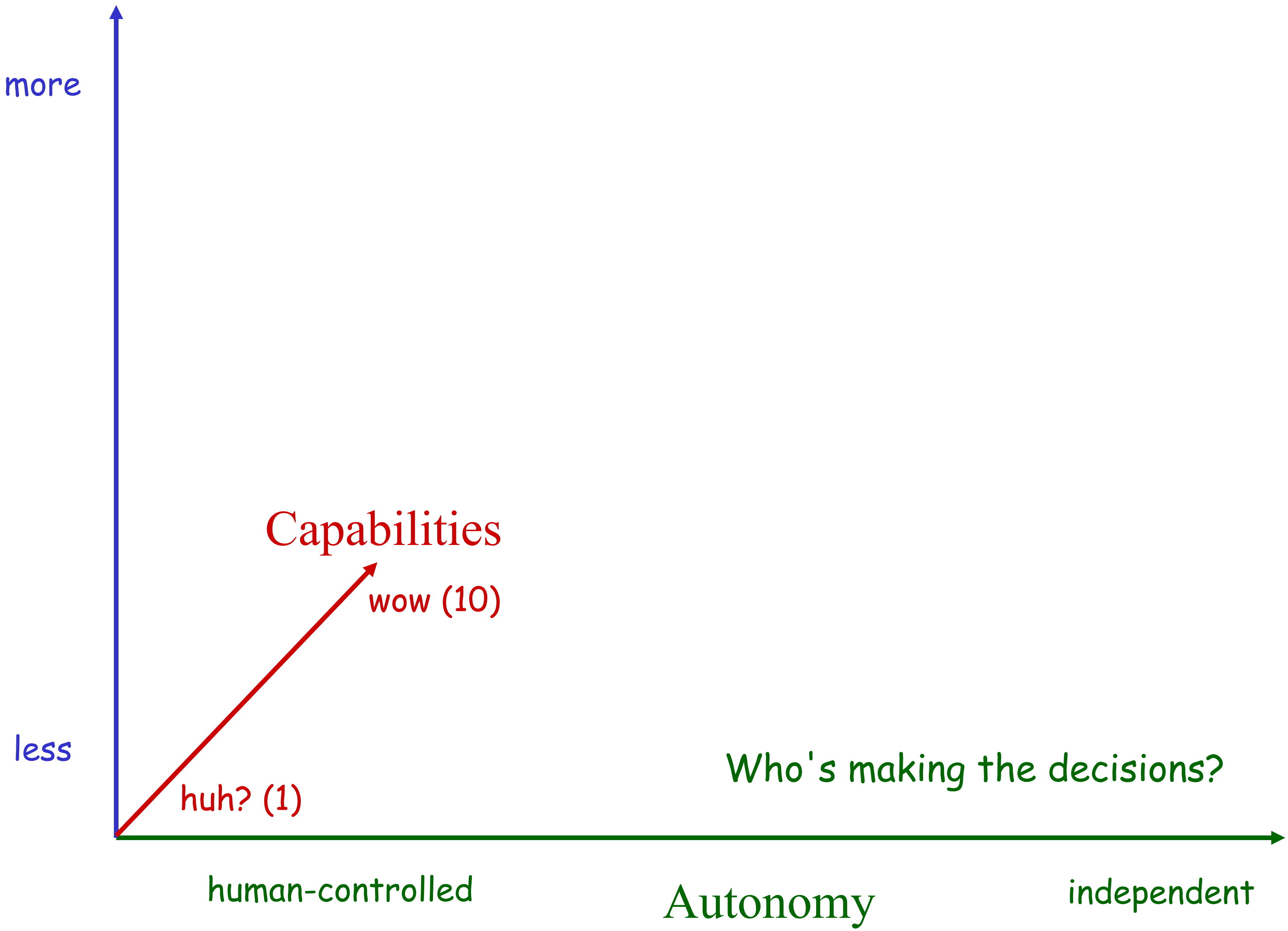
DARPA Manipulation Challenge

Robocup

There may be other axes along which to evaluate robots, too…

World Modeling

How much information about the world does the robot *internalize*?



Robot Plot

Capability (0-10)

World Modeling

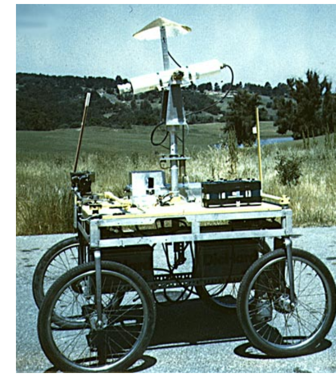
more

Al Gore (11)

Sims (5)

Stanley/Boss (9)

Shakey (3)



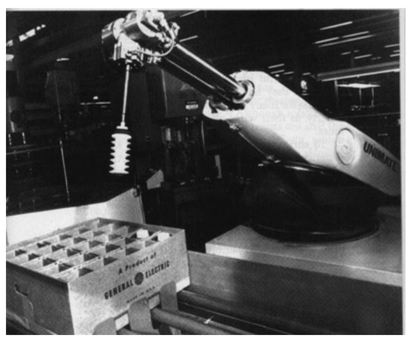
Stanford Cart (3)

MERs (8)

Bar Monkey (9)

less

da Vinci (2)



Unimate (4)

Roomba (7)

Genghis (3)

human-controlled

Autonomy

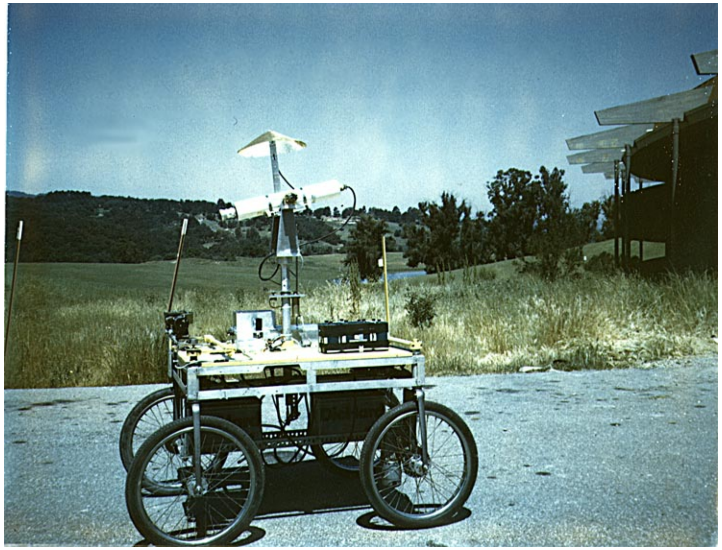
CS 154: **algorithms** for programming autonomous robots

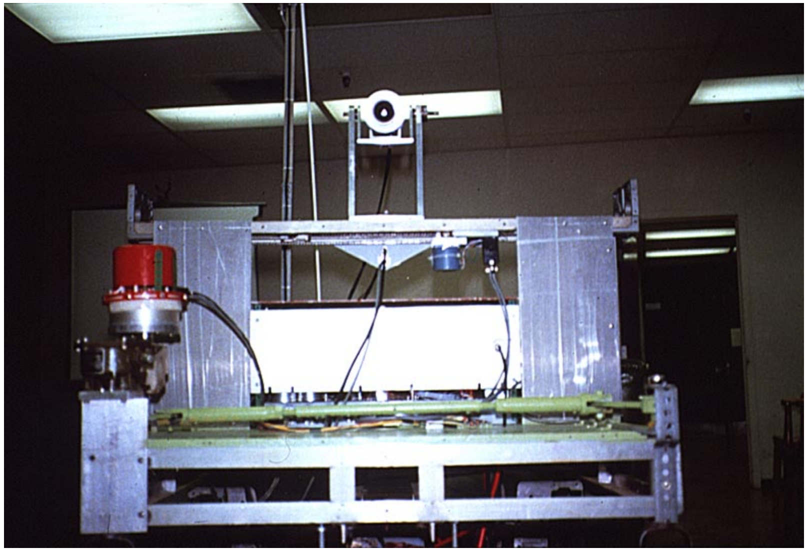
Sense Plan Act (SPA) Stanford Cart

Hans Moravec @ SAIL

"functional" task decomposition → SENSING | perception | world modeling | Planning | task execution | motor control | ACTING

"horizontal" subtasks





… 1976 …

Cartland (outdoors)

Reactive, Subsumption “Robot Insects”

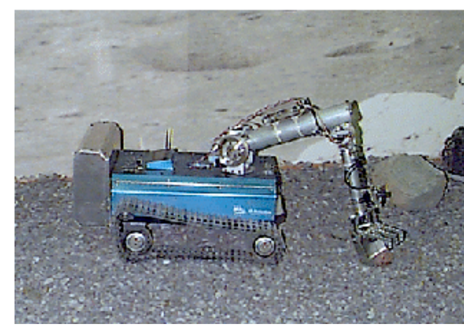
Rodney Brooks @ MIT

“behavioral” task decomposition →
“vertical” subtasks

SENSING

planning and reasoning
identify objects
build maps
explore
wander
avoid objects

ACTING



• • • 1985 • • •

Subsumption

Genghis

1) *Standing* by tuning the parameters of two behaviors:
the leg "swing" and the leg "lift"

2) *Simple walking*: one leg at a time

3) *Force Balancing*: via incorporated force sensors on the legs

4) *Obstacle traversal*: the legs should lift much higher if need be

5) *Anticipation*: uses touch sensors (whiskers) to detect obstacles

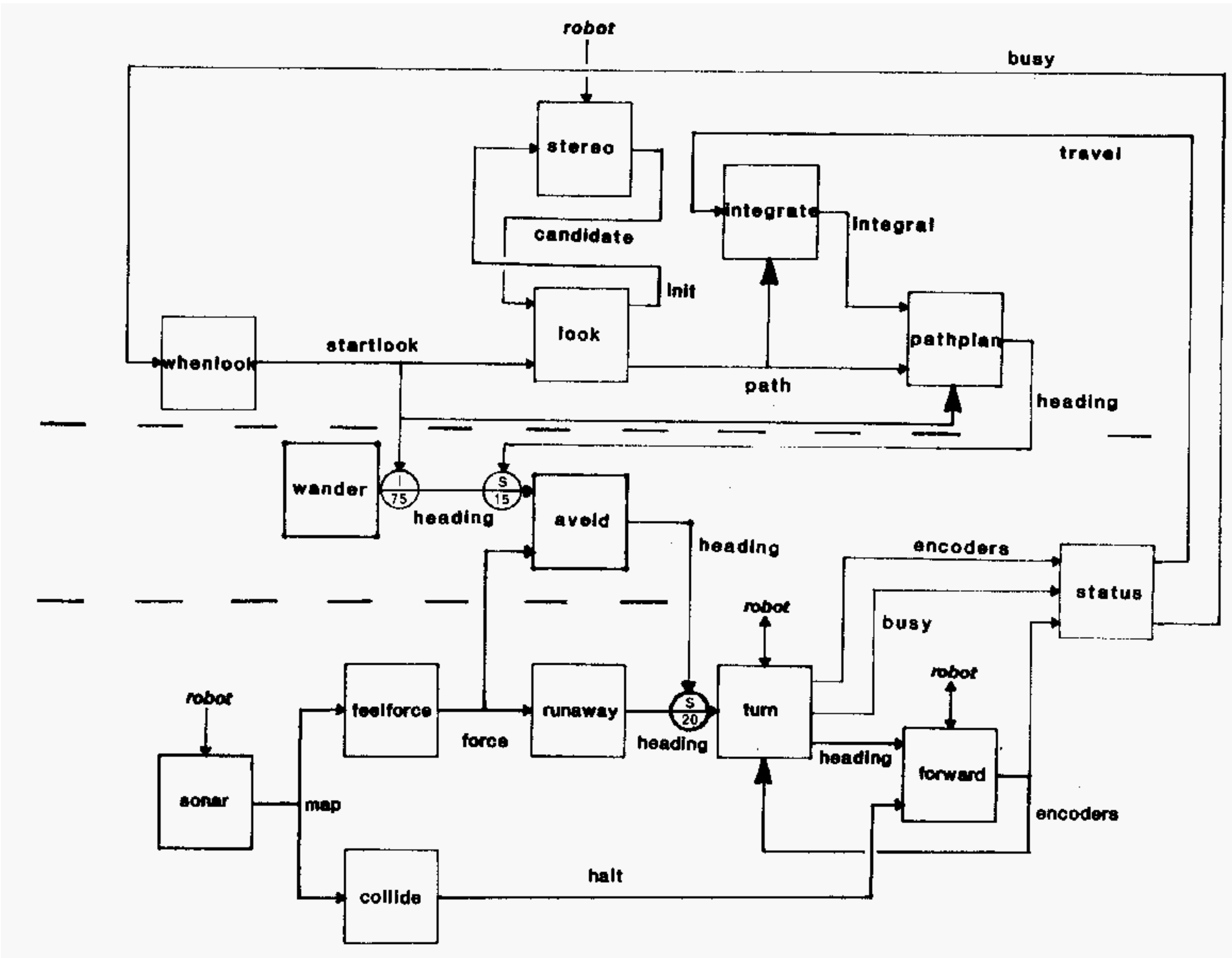
6) *Pitch stabilization*: uses an inclinometer to stabilize fore/aft pitch

7) *Prowling*: uses infrared sensors to start walking when a human approaches

8) *Steering*: uses the difference in two IR sensors to follow

57 modules wired together !

Subsumption Architecture



navigate behavior

wander behavior

runaway behavior

Robot Architecture

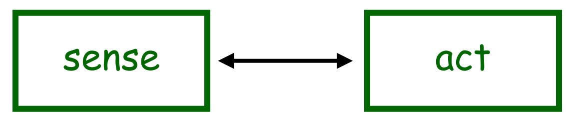
how much / how do we represent the world internally ?

As much as possible!

SPA paradigm

Not at all

Reactive paradigm



stimulus - response == "behavior"

Task-specific

Behavior-based architecture →
- **Subsumption paradigm**
- Potential Fields (later)

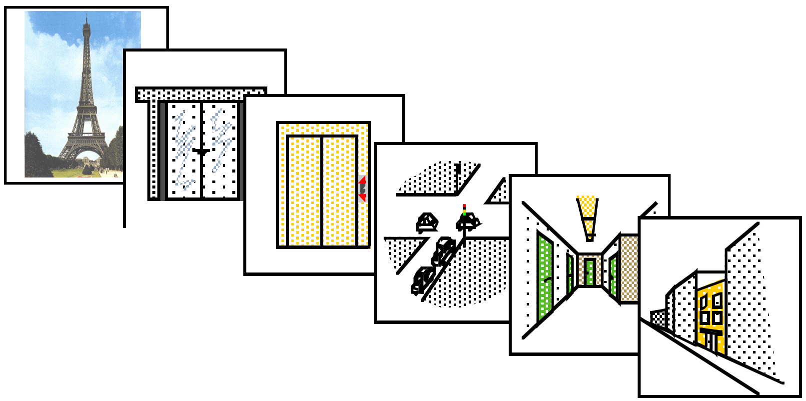
different ways of composing behaviors

As much as *possible.*

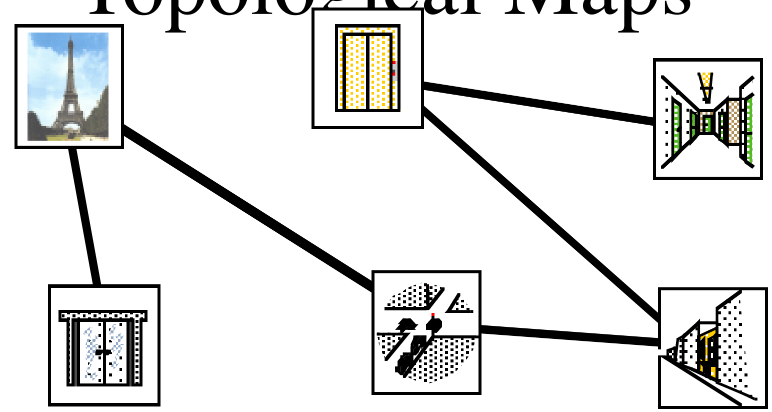
Hybrid approaches

Environment Representation: The Map Categories

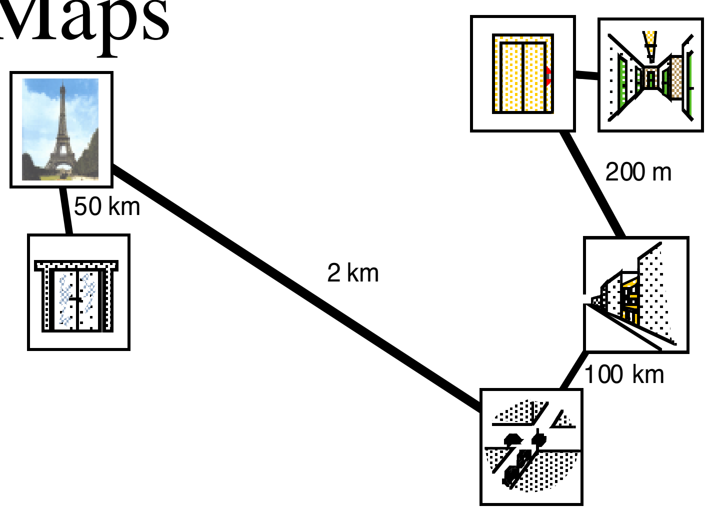
- Recognizable Locations



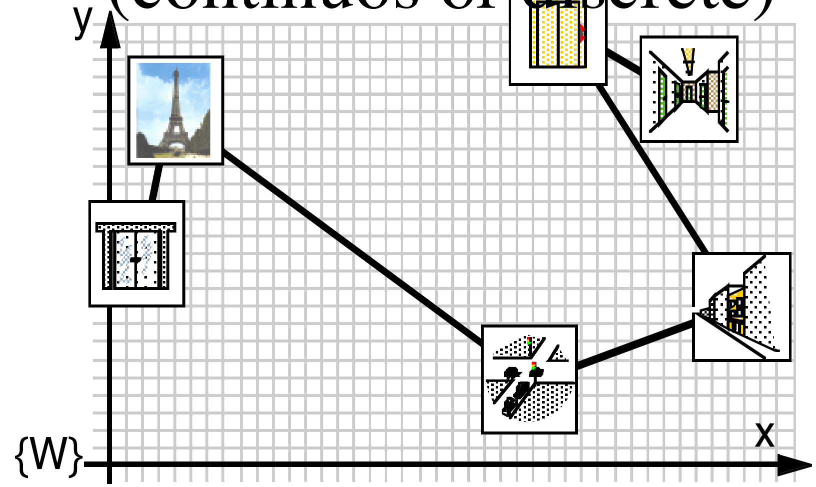
- Topological Maps



- Metric Topological Maps



- Fully Metric Maps (continuos or discrete)



Courtesy K. Arras

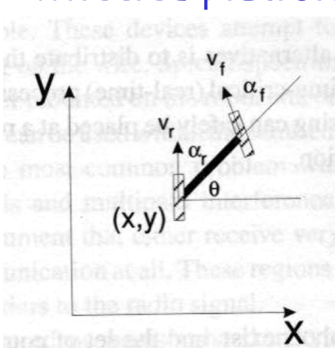
Kinematics

If we move a wheel one radian, how does the robot center move?

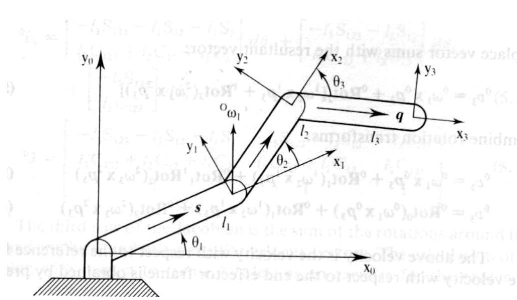
Robot geometry and kinematic analysis gives answers!

from sort of simple to
sort of complex

wheeled platforms

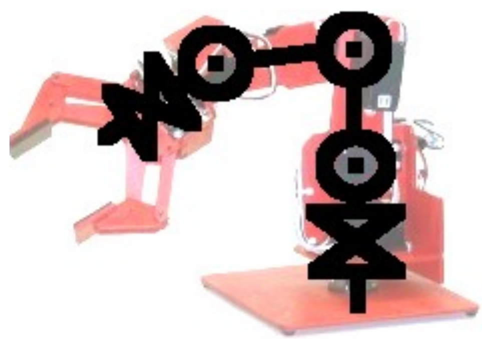


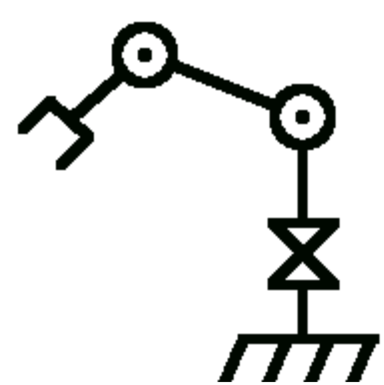
manipulator modeling



Kinematic modeling: Arm

Robot





Abstract model

Strategy:

1. Model each joint separately
2. Combine joints and linkage lengths

A simple example follows here.

More general treatment of next lecture

http://www.societyofrobots.com/robot_arm_tutorial.shtml

Coordinate frames & forward kinematics

- Three coordinate frames: 0 1 2
- Positions:

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} a_1 \cos(\theta_1) \\ a_1 \sin(\theta_1) \end{bmatrix}$$

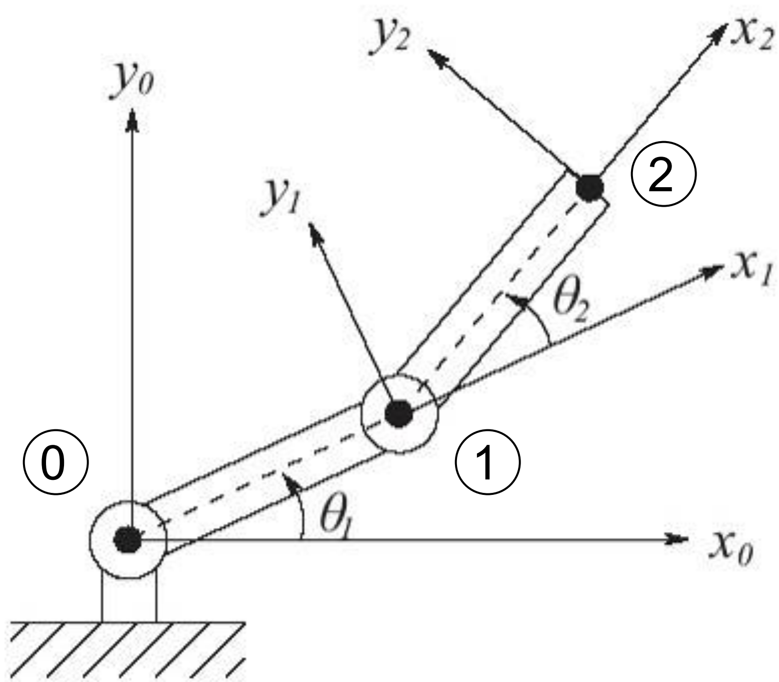
$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} a_1 \cos(\theta_1) + a_2 \cos(\theta_1 + \theta_2) \\ a_1 \sin(\theta_1) + a_2 \sin(\theta_1 + \theta_2) \end{bmatrix} \equiv \begin{bmatrix} x \\ y \end{bmatrix}_t$$

- Orientation of the tool frame:

$$\hat{x}_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \hat{y}_0 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\hat{x}_2 = \begin{bmatrix} \cos(\theta_1 + \theta_2) \\ \sin(\theta_1 + \theta_2) \end{bmatrix}, \hat{y}_2 = \begin{bmatrix} -\sin(\theta_1 + \theta_2) \\ \cos(\theta_1 + \theta_2) \end{bmatrix}$$

$$R_2^0 = \begin{bmatrix} \hat{x}_2 \cdot \hat{x}_0 & \hat{y}_2 \cdot \hat{x}_0 \\ \hat{x}_2 \cdot \hat{y}_0 & \hat{y}_2 \cdot \hat{y}_0 \end{bmatrix} = \begin{bmatrix} \cos(\theta_1 + \theta_2) & -\sin(\theta_1 + \theta_2) \\ \sin(\theta_1 + \theta_2) & \cos(\theta_1 + \theta_2) \end{bmatrix}$$

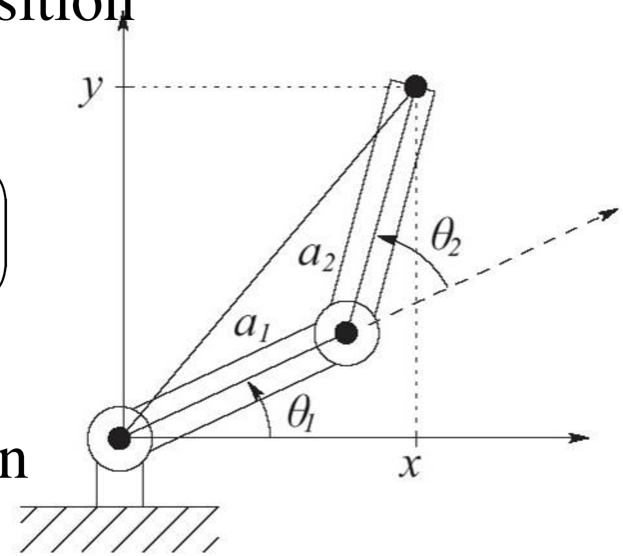


Inverse kinematics

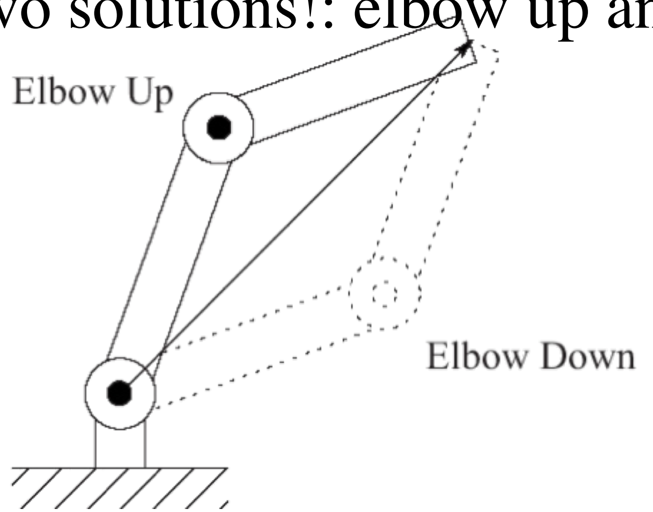
- Find the joint angles for a desired tool position

$$\cos(\theta_2) = \frac{x_t^2 + y_t^2 - a_1^2 - a_2^2}{2a_1a_2} \equiv D \Rightarrow \sin(\theta_2) = \pm\sqrt{1-D^2}$$

$$\theta_2 = \tan^{-1}\left(\pm\frac{\sqrt{1-D^2}}{D}\right) \quad \theta_1 = \tan^{-1}\left(\frac{y}{x}\right) - \tan^{-1}\left(\frac{a_2\sin(\theta_2)}{a_1 + a_2\cos(\theta_2)}\right)$$

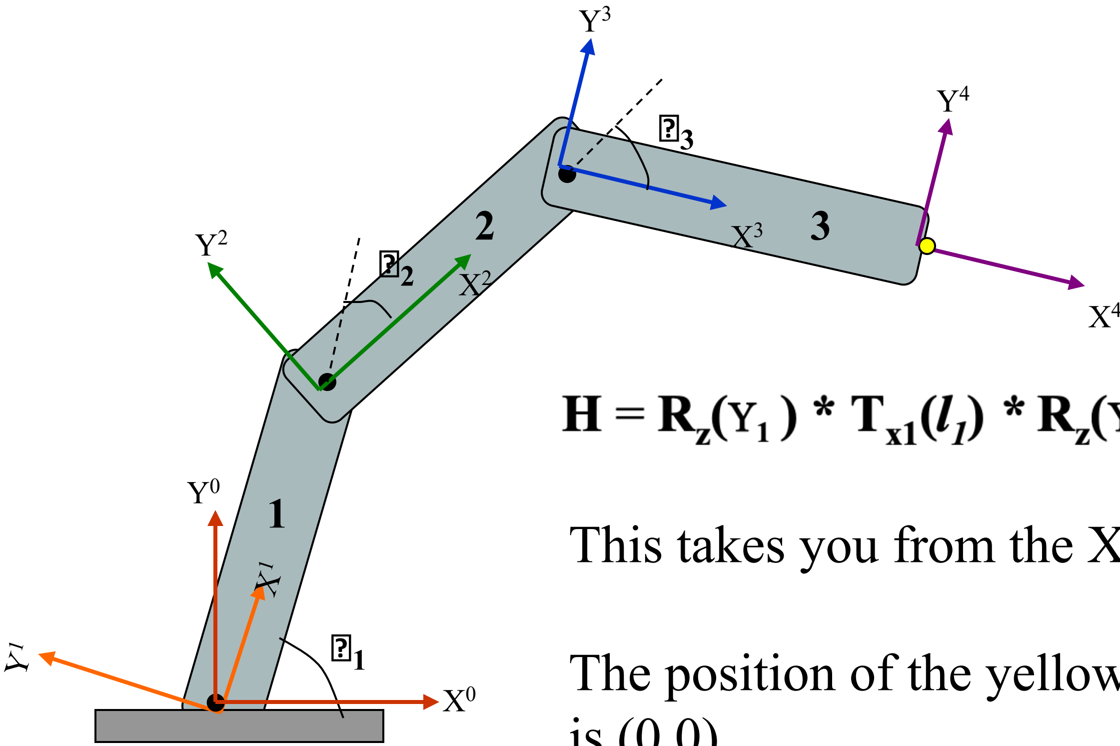


- Two solutions!: elbow up and elbow down



Homogenous coordinates: Uniform matrix representation of translation and rotation

The yellow dot is the origin of a tool coordinate X^4Y^4 frame



$$\mathbf{H} = \mathbf{R_z}(\mathrm{Y}_1) * \mathbf{T_{x1}}(\boldsymbol{l_1}) * \mathbf{R_z}(\mathrm{Y}_2) * \mathbf{T_{x2}}(\boldsymbol{l_2}) * \mathbf{R_z}(\mathrm{Y}_3) * \mathbf{T_{x3}}(\boldsymbol{l_3})$$

This takes you from the X^0Y^0 frame to the X^4Y^4 frame.

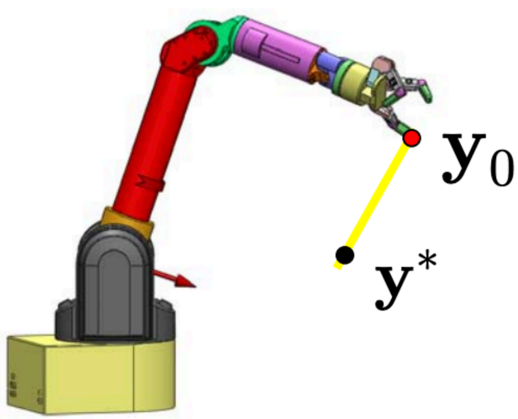
The position of the yellow dot relative to the X^4Y^4 frame is (0,0).

$$\begin{bmatrix} \mathbf{X} \\ \mathbf{Y} \\ \mathbf{Z} \\ \mathbf{1} \end{bmatrix} = \mathbf{H} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix}$$

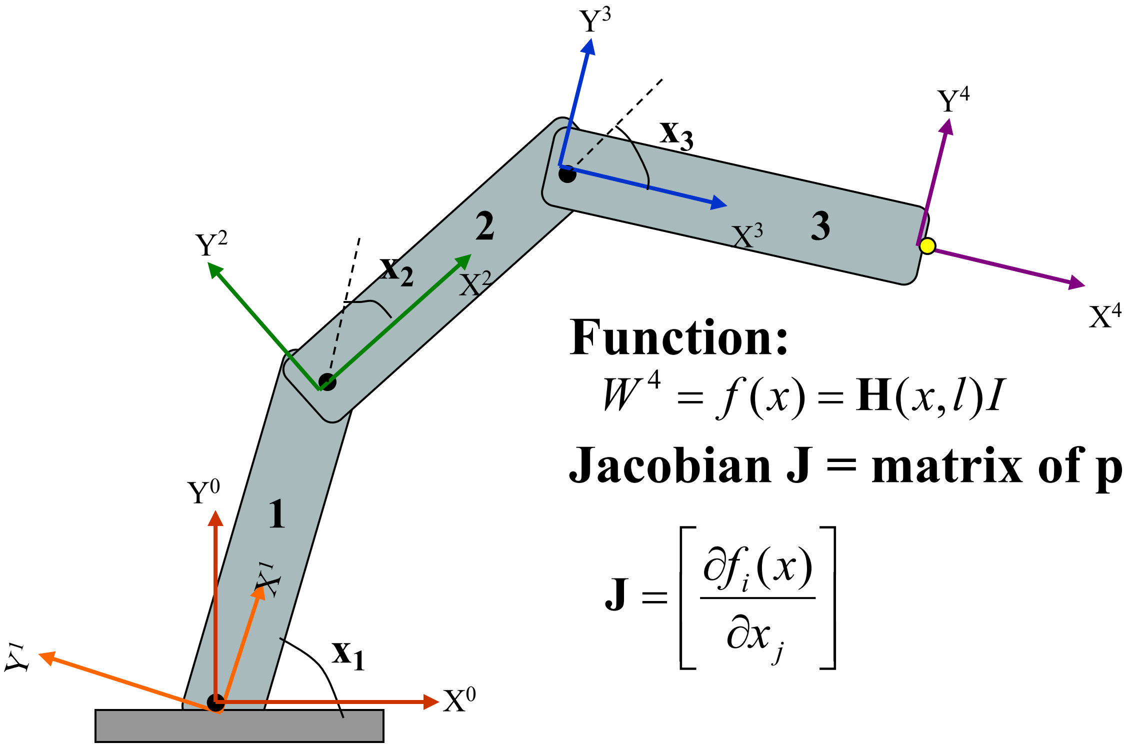
Notice that multiplying by the (0,0,0,1) vector will equal the last column of the H matrix.

Numerical Inverse Kinematics

- Cartesian Location $\mathbf{y} = [x, y, z]^T \quad = \quad f(\mathrm{x})$
- Motor joint angles: $\mathbf{x} = [\, x_1, \quad x_2, \ldots \quad x_n \,]^T$
- Local linear model: $\Delta \mathbf{y} = \mathbf{J}(\mathbf{x}) \Delta \mathbf{x}$
- Numerical steps: 1 Solve: $\mathbf{y}^* - \mathbf{y}_k = \mathbf{J} \Delta \mathbf{x}$
  2 Update: $\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta \mathbf{x}$



The Numeric solution: How to solve? Newton's Method



Function:

$$W^4 = f(x) = \mathbf{H}(x,l)I$$

Jacobian J = matrix of partial derivatives:

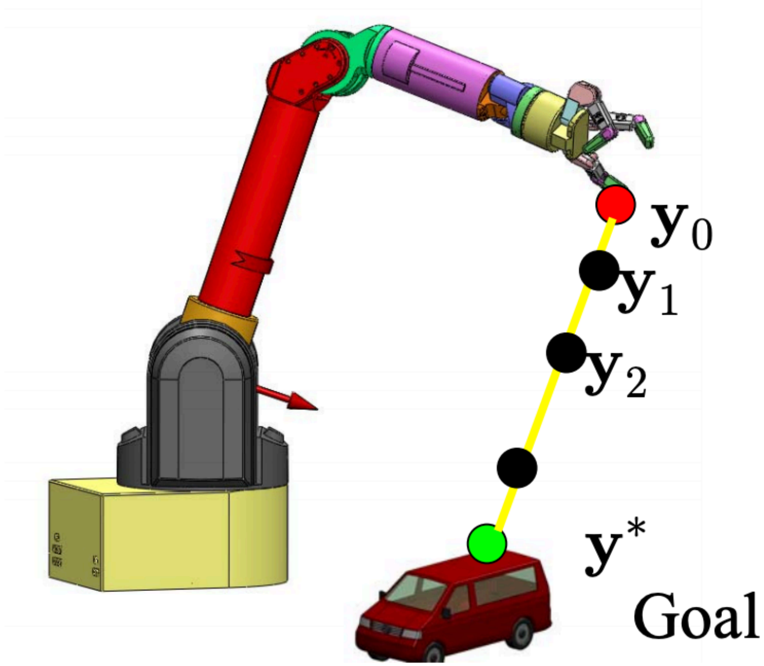
$$\mathbf{J} = \left[\frac{\partial f_i(x)}{\partial x_j} \right]$$

- **Analytic J: Take derivatives of H**
- **Numeric estimation: Test movements or Broydens method (cmput 340, Heath Ch 5,6)**

Numerical Inverse Kinematics

repeat

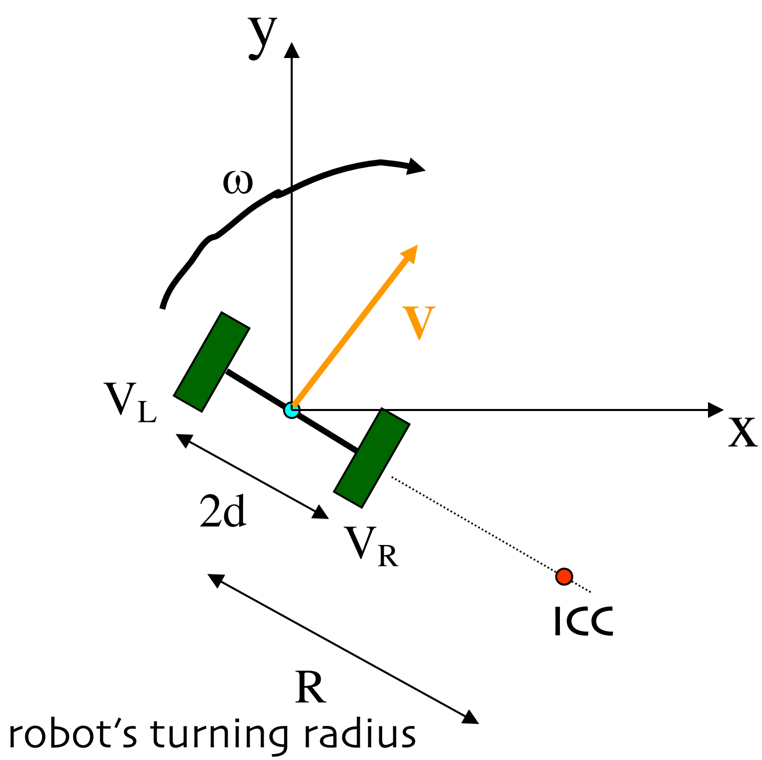
1. Solve for motion: $[\mathbf{y} \quad - \mathbf{y}_k] = \mathbf{J}\Delta\mathbf{x}$
2. Move robot joints: $\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x}$
3. Read actual Cartesian move $\Delta\mathbf{y}$
4. Update Jacobian: $\hat{J}_{k+1} = \hat{J}_k + \dfrac{(\Delta\mathbf{y} - \hat{J}_k\Delta\mathbf{x})\Delta\mathbf{x}^T}{\Delta\mathbf{x}^T\Delta\mathbf{x}}$



Move the robot to each subgoal in sequence $\mathbf{y}_k$

Iterate until convergence at final goal

Kinematics: Differential drive



1) Specify system measurements
   - consider possible coordinate systems

2) Determine the point (the radius) around which the robot is turning.

   - each wheel must be traveling at the same angular velocity **around the ICC**

3) Determine the robot's speed around the ICC and its linear velocity

$$\omega(R+d) = V_L$$
$$\omega(R-d) = V_R$$

Thus,

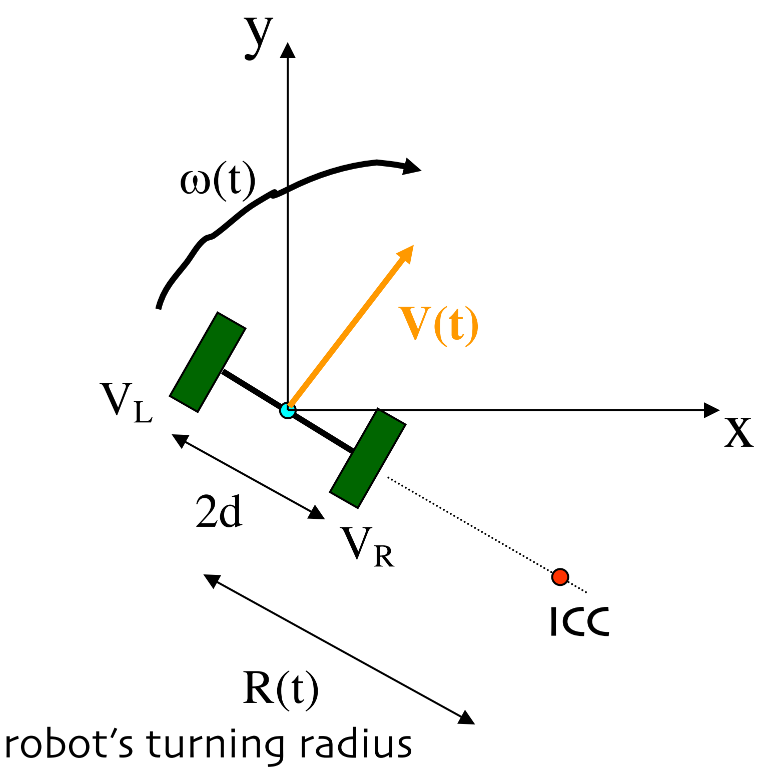
$$\omega = (V_R - V_L) / 2d$$

$$R = d\,(V_R + V_L) / (V_R - V_L)$$

So, where's the robot?

So, the robot's velocity is $V = \omega R = (V_R + V_L) / 2$

Kinematics: Differential drive



4) Integrate to obtain position

$$V_x = V(t)\cos(\theta(t))$$

$$V_y = V(t)\sin(\theta(t))$$

Thus,

$$x(t) = \int V(t)\cos(\theta(t))\,dt$$

$$y(t) = \int V(t)\sin(\theta(t))\,dt$$

$$\theta(t) = \int \omega(t)\,dt$$

Kinematics

with

$$\omega = (V_R - V_L) / 2d$$

$$R = d\,(V_R + V_L) / (V_R - V_L)$$

$$V = \omega R = (V_R + V_L) / 2$$

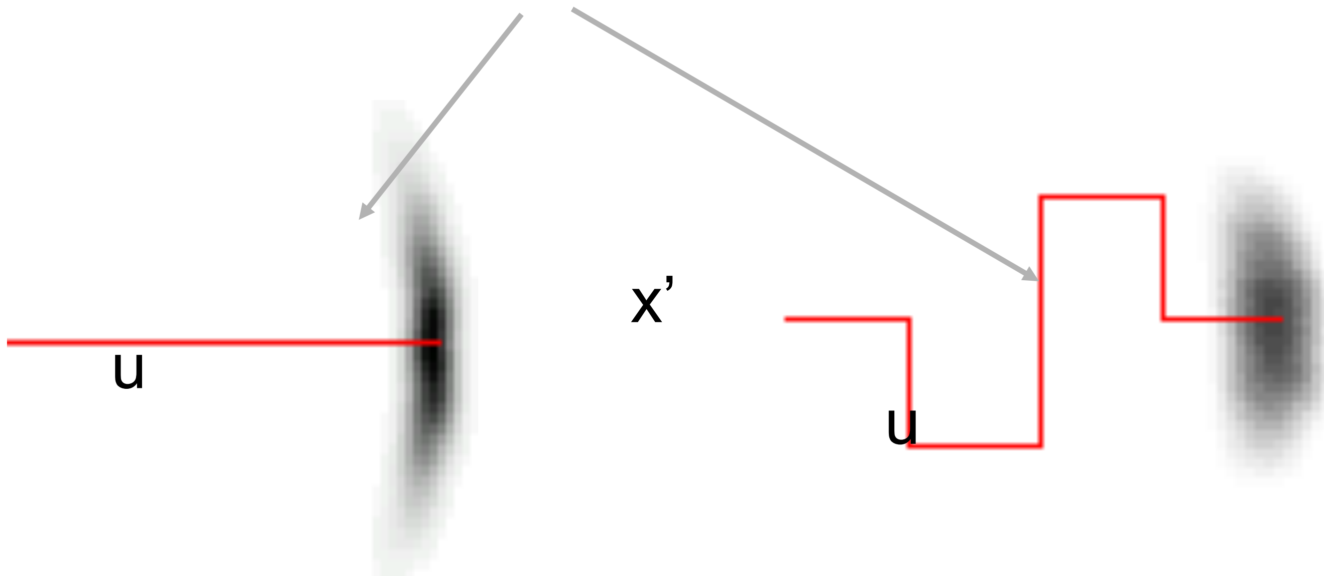
What has to happen to change the ICC ?

things have to change over time, **t**

Uncertain motion: Probabilistic kinematics

- Repeated application of the sensor model for short movements.
- For a motion command **u**, and starting position **x'**, what is the probability to end at **x**?

$$p(x|u,x')$$



holonomicity

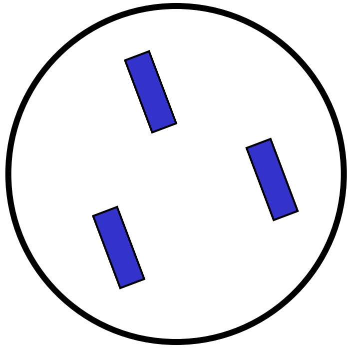
Mobile robots mentioned thus far share an important (if frustrating) property: they are **nonholonomic** .

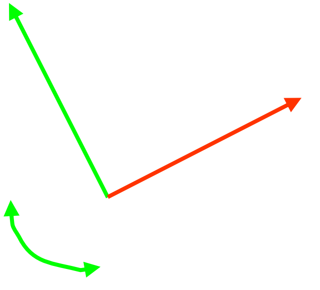
- makes it more difficult to navigate between two arbitrary points

- need to resort to techniques like parallel parking

By definition, a robot is **holonomic** if it *can* move to change its pose instantaneously in all available directions.

But the Nomad's differential motion ***is*** constrained.





Synchro Drive

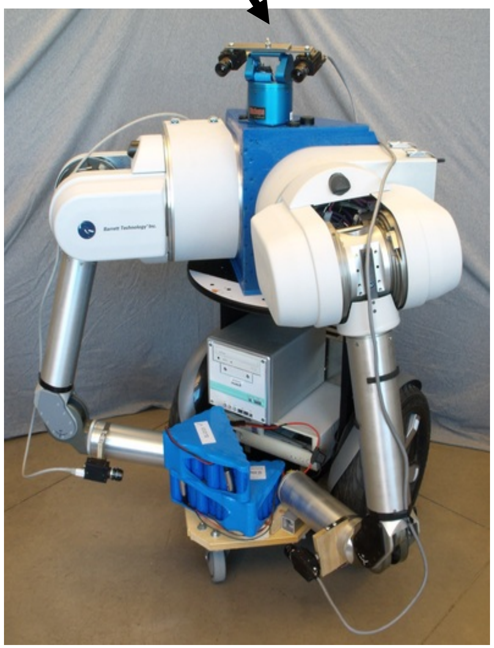
two DOF are freely controllable; the third is only indirectly accessible

Sensors:

- Cameras

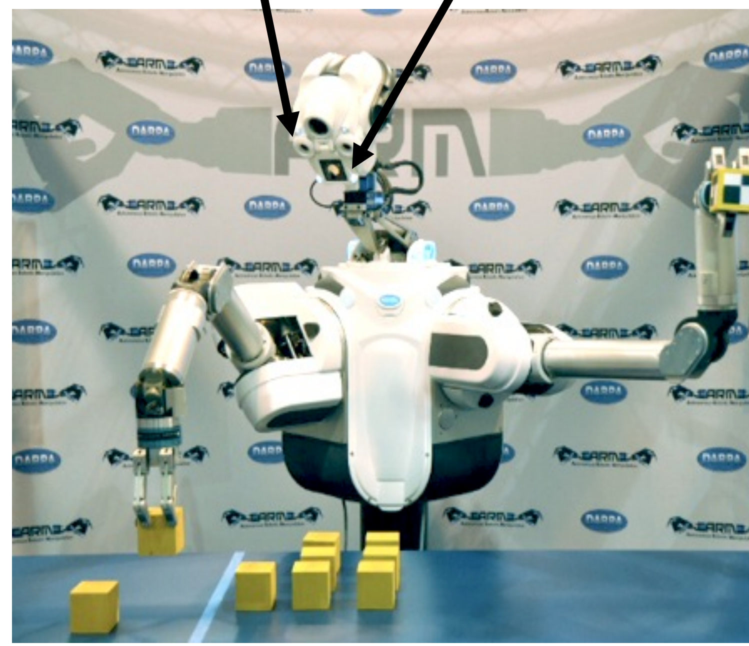
Laser range finder

Cameras and 3D TOF laser



Our robot

CMU Herb



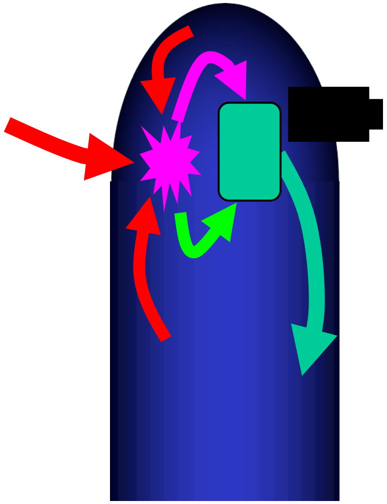
Darpa Manip challenge

Sensors on a $300 UAV

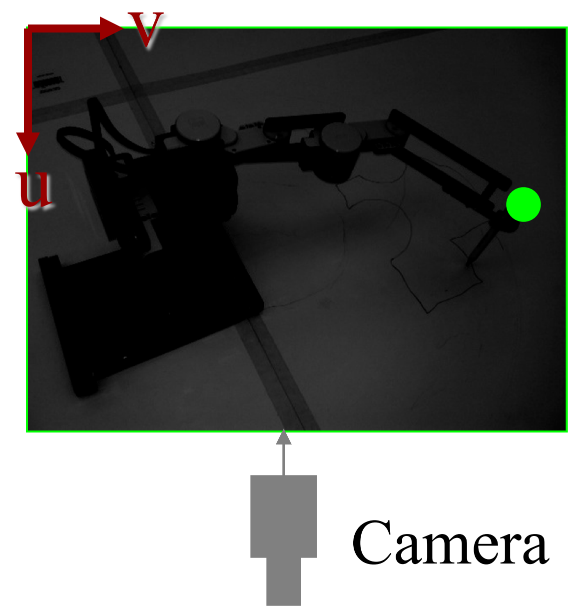
- AR.Drone

Classification of Sensors

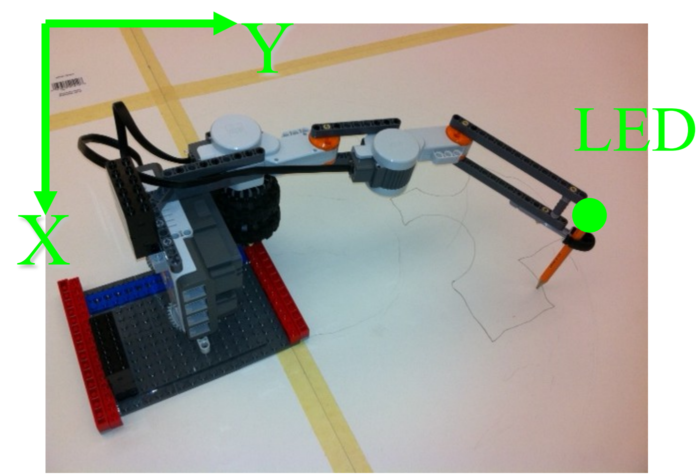
- Where is the information coming from?
  - Inside: Proprioceptive sensors
    - motor speed, wheel load, heading of the robot, battery status
  - Outside: Exteroceptive sensors
    - distances to objects, intensity of the ambient light, unique features
- How does it work? Requires energy emission?
  - No: Passive sensors
    - temperature probes, microphones, CCD
  - Yes: Active sensors
    - Controlled interaction -> better performance
    - Interference
- Simple vs. composite (sonar vs. wheel sensor)



Simple visual sensing: LED markers



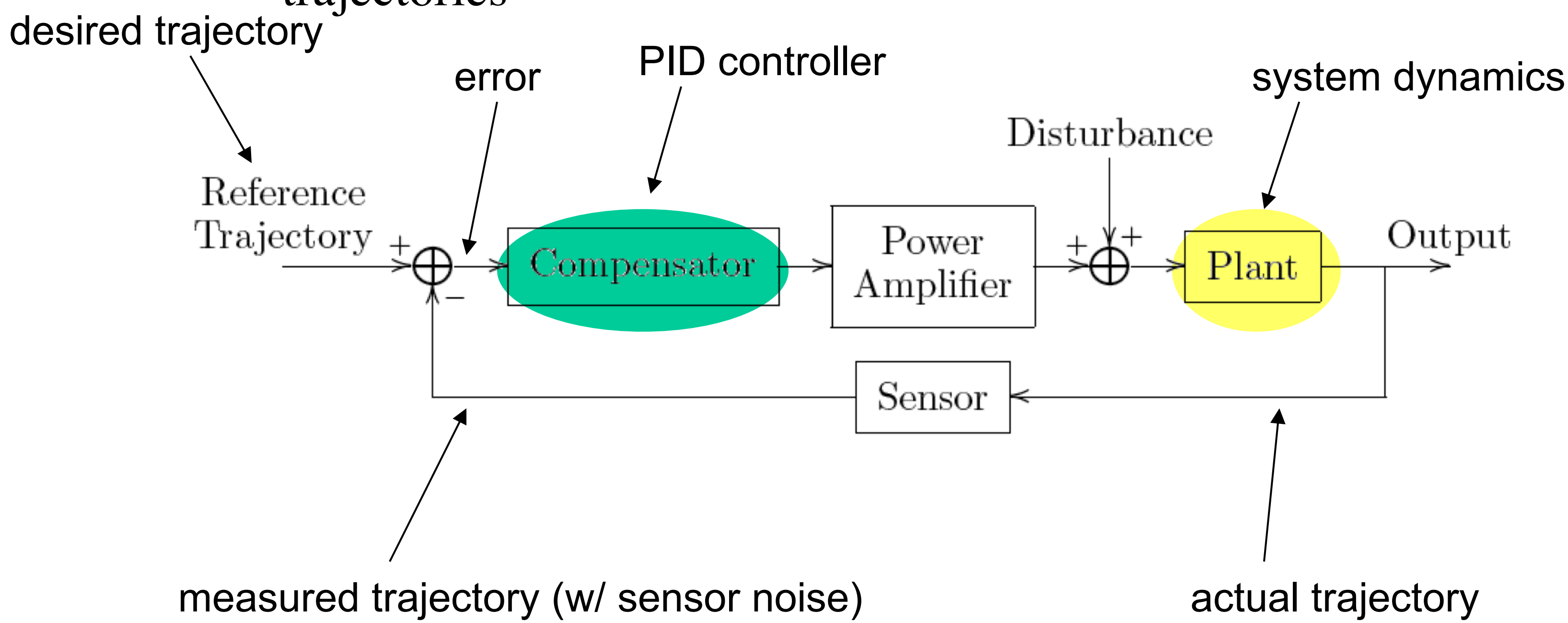
- Put camera overhead pointing straight down on worktable.
  - Adjust cam position so pixel [u,v] = s[X,Y]. Pixel coordinates are scaled world coord
  - Lower brightness so LED brightest
- Put LED on robot end-effector
- Detection algorithm:
  - Threshold brightest pixels I(u,v)>200
  - Find centroid [u,v] of max pixels
- Variations:
  - Blinking LED can enhance detection in ambient light.
  - Different color LED’s can be detected separately from R,G,B color video.



Robot motor (joint) control

(More of an Engineering discipline than CS)

- We can create a desired tool path/velocity
  - Use inverse kinematics and Jacobian to create desired joint trajectories



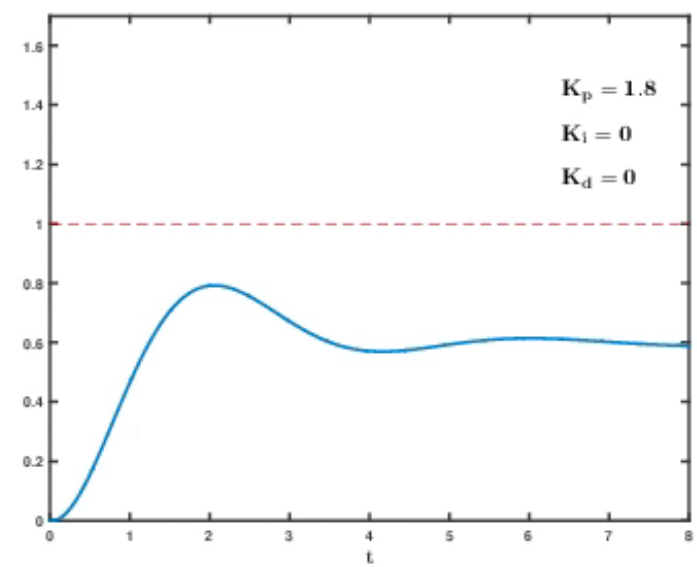
PID control

- **P**roportional + **I**ntegral + **D**erivative Control

$$u(t) = K_p\, e(t) + K_i \int_0^t e(\tau) d\tau + K_d\, \dot{e}(t)$$

Effects of *increasing* a parameter independently[20][21]

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



[wikipedia: PID control]

PID: digital implementation

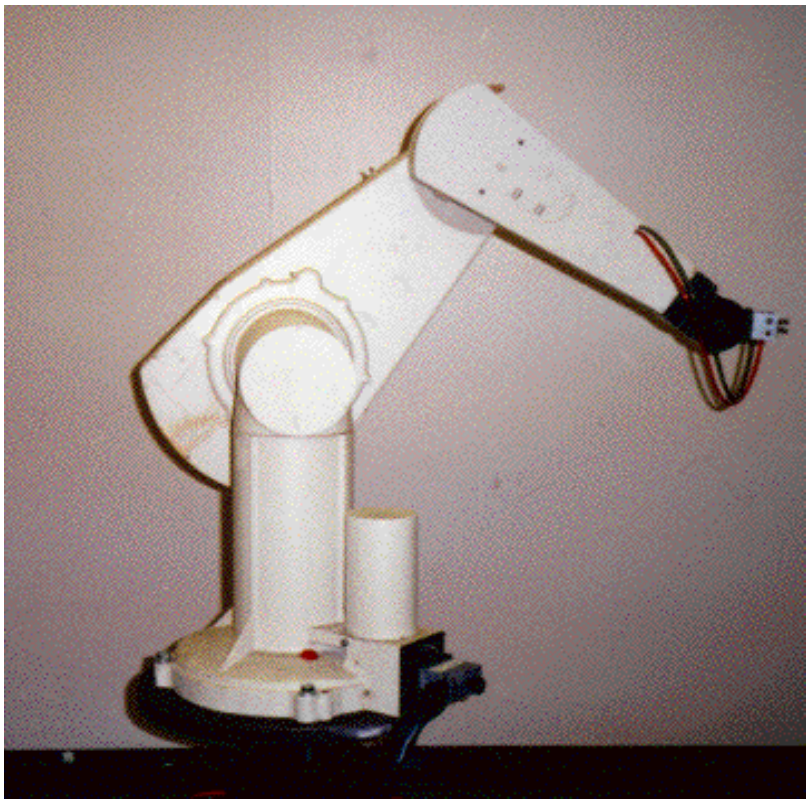
- Discrete integral $\int_0^t e(\tau)d\tau = \sum_{i=1}^{k} e(t_i)\Delta T$
- Derivative (backward finite difference) $\dot{e}(t) = \frac{e_{t_k} - e_{t_{k-1}}}{\Delta T}$

Pseudocode

```
previous_error = 0
integral = 0
loop:
  error = setpoint - measured_value
  integral = integral + error*dt
  derivative = (error - previous_error)/dt
  output = Kp*error + Ki*integral + Kd*derivative
  previous_error = error
  wait(dt)
  goto loop
```

Accuracy, Repeatability and Resolution

An classic arm - The PUMA 560



- Accuracy
  - Factory about 10mm
  - Calibrated 1-5mm
- Repeatability 1mm
- Resolution 0.1mm

Upcoming: More accurate control though Visual Servoing