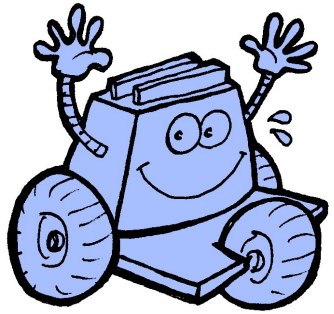
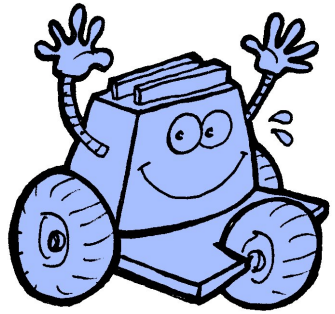
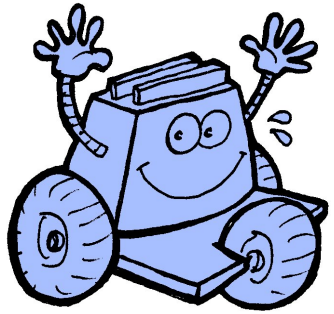


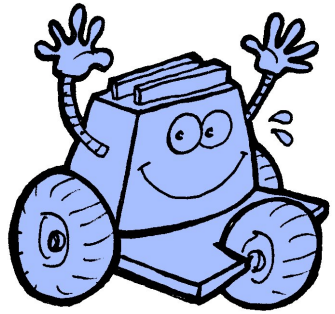


B31YS
Robotics Systems Science

Or how to learn how to play with robots quickly…

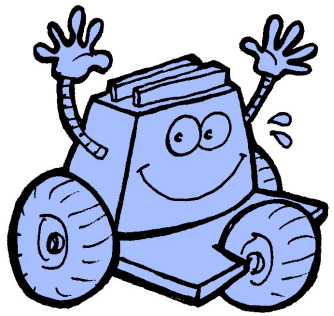


Course Overview

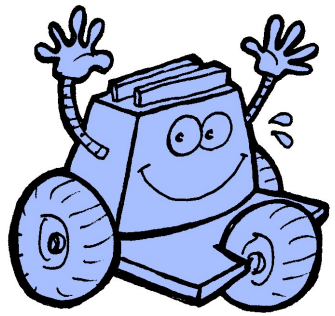


- The aim of the course is to present a unified view of the field, through experimentation in simulation and culminating in a practical involving the development of an integrated robotic system that actually embodies key elements of the major algorithmic techniques.

- **Syllabus:**
- Robotics Operating Systems
- Motion planning - basic and sampling based methods
- State estimation, localization and mapping
- Implementing SLAM; Multi-modal sensor fusion
- Two-view geometry
- Mission and task planning
- Middleware and software engineering for robot systems



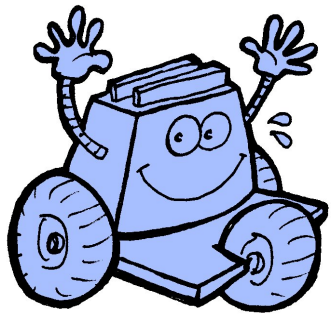
Course Overview



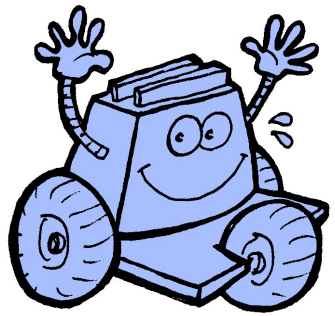
The course will require some knowledge of the following fundamental notions: Multivariate Calculus, Linear Algebra and matrix manipulations, Basic notions of Statistics and concepts including expectation and conditional probability. General programming competence is assumed. The course will use C++ / python in a Linux environment, GIT and OpenCV.

- **Reading list:**
- **S. Thrun, W. Burgard and D. Fox, Probabilistic Robotics.**
- **Peter Corke, Robotics Vision and Control, Springer.**
- **Richard Szeliski, Computer Vision: Algorithms and Applications, pdf available online.**
- **Bruno Siciliano & al: Robotics: Modelling, Planing and Control, Springer.**
- **Efforts:** 150 (Lecture Hours 30, Supervised Practical/Workshop/Studio Hours 30, Summative Assessment Hours 2, Directed Learning and Independent Learning Hours 88. You should expect to spend approximately 40 hours on the coursework for this course.
-

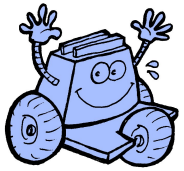
- **Timetable**
- Lectures: Monday 10h15-11h15 EM 3.02, Monday 16h15-17h15 EM 2.23, Thursday 16h15-17h15 EM 3.02
- Robotics Labs: Monday and Friday 13h15-15h15 EM 2.23
- Personal Work: We expect student on the course to spend a minimum of 5 hours per week of personal work.



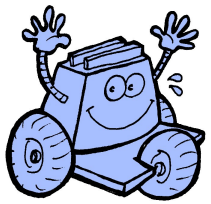
Key remarks on course



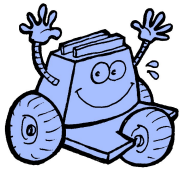
- It is ambitious
- It is a first, so expect problems and give us feedback
- It is learning by doing – spend the time learning and exploring by yourselves
- Use Ros-Ignite as much as you can to test ideas and learn new topics
- Enjoy yourselves
- Try to spend 15 hours per week on the course!x



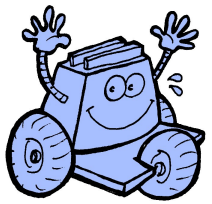
Course timeline



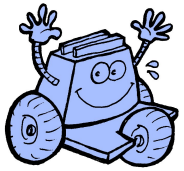
Week	Activity (academic lead)	Expected outcome from teams (student lead)	Lead academic
1	Introduction to module. Initial introduction to turtlebot and ROS	1. Watch Videos about ROS provided (ETH) 2. Install ROS image (provided) on your personal laptop. 3. Do course ROS in a week from ROS ignite and complete first task (see separate sheet).	Yvan Petillot Helmi Fraser
2	Basic Motion Planning and ROS	1. Complete ROS in a week.. 2. Demonstrate simple motion planning (W2 assessment) 3. Attend ROS Lab Session	Yvan Petillot Helmi Fraser
3 & 4	Filtering and Localisation Probabilistic robotics	1. 3 hours lab session (matlab for KF) 2. AMCL in simulation using Laser + odometry. 3. Demonstration of one method (EKF/PF) on rosbags – odometry + GPS - 5% of final mark	Sen Wang
5 & 6	Introduction to computer vision	1. Course supported by QUT course (3 hours). Image formation, Perspective geometry, Epipolar geometry, stereo vision, visual odometry, feature extraction, OpenCV. Presentation by students. Feedback provided. 2. Course (3stereo hours). Image formation, Perspective geometry, Epipolar geometry, vision, structure from motion, feature extraction, OpenCV. 3. Camera calibration lab 4. Demonstration of assignment in real robot or simulator – 5% of final mark	Tomasz Luczynski



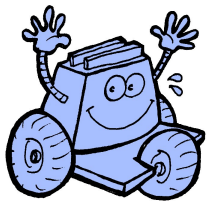
Course timeline



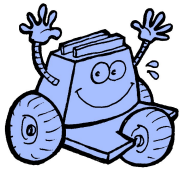
Week	Activity (academic lead)	Expected outcome from teams (student lead)	Lead academic
7	Visual Slam	1. EKF SLAM, Graph SLAM, Loop Closure with bags of words 2. Implementation of one existing implementation of visual slam in ROS on turtlebot. - 5% of final mark	Sen Wang
8	Task planning	1. Student presentations on path planning (DWA, Potential Fields, A*, RRTs) 2. Task / Mission planning 3. Final Project selection 4. Presentation on choice of SLAM algorithm from week 7. 5% of final mark	Yvan Petillot Yaniel Carreno
9-12	Project work	Can be proposed by students, 30 % of final mark. Has to use simulation and a real robot	ALL
12-14	Exam	2 hours, 50% of final mark	



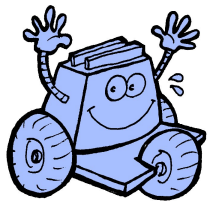
Schedule Week 1 & 2



	Monday			Thursday	Friday
	1015-1115	1315-1515	1615-1715	1615-1715	1315-1515
Week 1	Lecture: Introduction to course and first 2 weeks First ROS Lecture	Lab: Installation of ROS on machines and Introduction to ROS ignite environment	Lecture: ROS continued	Lecture: Introduction to reactive motion planning	Lab: ROS surgery
Week 2	Introduction to assignment 1 ROS Surgery	Lab: Support for ROS in a week and assignment 1	Lab: Support for ROS in a week and assignment 1	Lab: Support for ROS in a week and assignment 1	Demonstration of Assignment 1 in simulation.

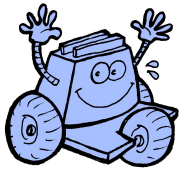


Evaluation



- 4 assignments @ 5% each
- 1 3-5 weeks project @30%
- 1 Exam (2 hours) @ 50%

?



What are robots?

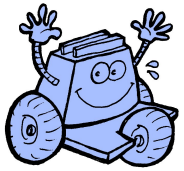
- Robota (Czech) = A worker of forced labor

  From Czech playwright Karel Capek's 1921 play “R.U.R” (“Rossum's Universal Robots”)

- Japanese Industrial Robot Association (JIRA) :

  “A device with degrees of freedom that can be controlled.”

  - Class 1 : Manual handling device
  - Class 2 : Fixed sequence robot
  - Class 3 : Variable sequence robot
  - Class 4 : Playback robot
  - Class 5 : Numerical control robot
  - Class 6 : Intelligent robot

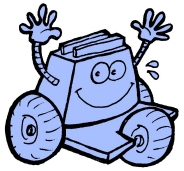


A Brief History of Robotics

- Mechanical Automata
  - Ancient Greece & Egypt
    - Water powered for ceremonies
  - 14^{th} – 19^{th} century Europe
    - Clockwork driven for entertainment
- Motor driven Robots
  - 1928: First motor driven automata
  - 1961: Unimate
    - First industrial robot
  - 1967: Shakey
    - Autonomous mobile research robot
  - 1969: Stanford Arm
    - Dextrous, electric motor driven robot arm

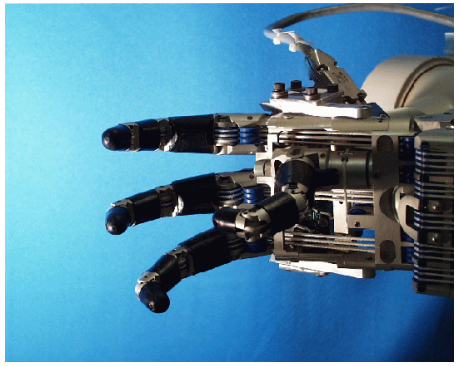
Maillardet's Automaton

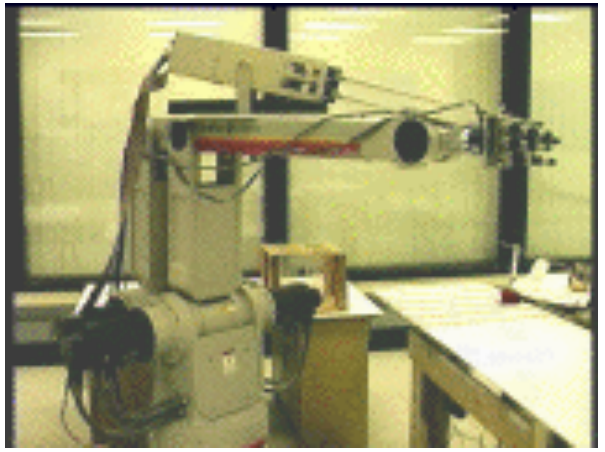
Unimate



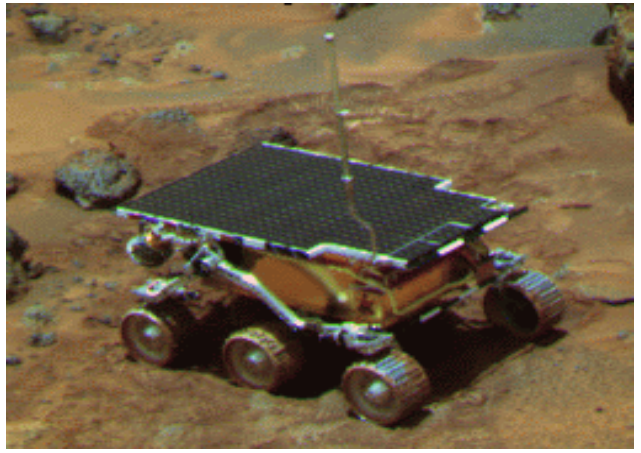
Robot Types

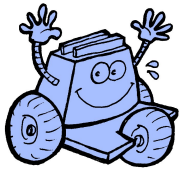
- Robot Manipulators





- Mobile Robots





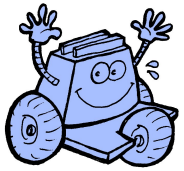
Robot types

- Walking Robots

@Edinburgh

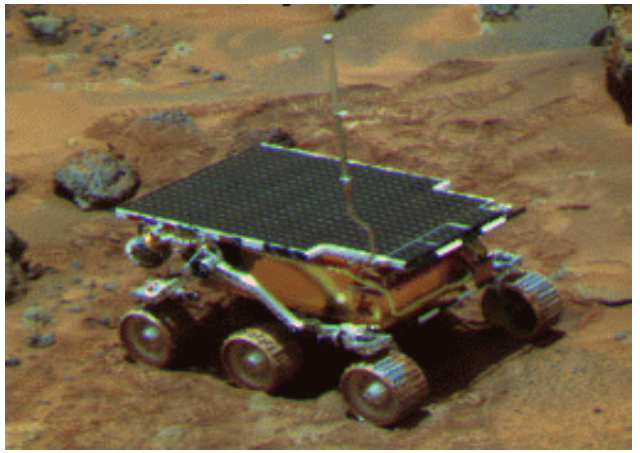
@Edinburgh

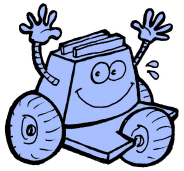
- Humanoid Robots



Autonomous Robots

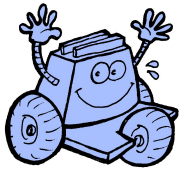
- The control of autonomous robots involves a number of subtasks
  - Understanding and modeling of the mechanism
    - Kinematics, Dynamics, and Odometry
  - Reliable control of the actuators
    - Closed-loop control
  - Generation of task-specific motions
    - Path planning
  - Integration of sensors
    - Selection and interfacing of various types of sensors
  - Coping with noise and uncertainty
    - Filtering of sensor noise and actuator uncertainty
  - Creation of flexible control policies
    - Control has to deal with new situations





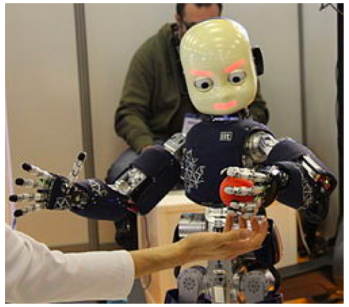
Traditional Industrial Robots

- Traditional industrial robot control uses robot arms and largely pre-computed motions
  - Programming using “teach box”
  - Repetitive tasks
  - High speed
  - Few sensing operations
  - High precision movements
  - Pre-planned trajectories and task policies
  - No interaction with humans

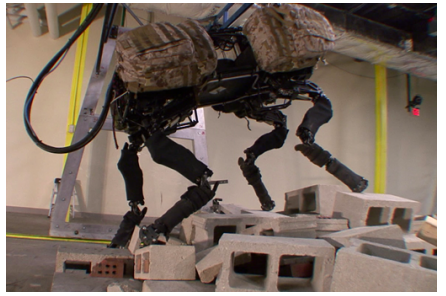


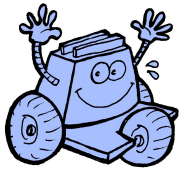
Problems

- Traditional programming techniques for industrial robots lack key capabilities necessary in intelligent environments
  - Only limited on-line sensing
  - No incorporation of uncertainty
  - No interaction with humans
  - Reliance on perfect task information
  - Complete re-programming for new tasks



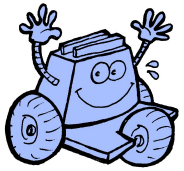
@HWU





Requirements for Autonomous Robots in unstructured Environments

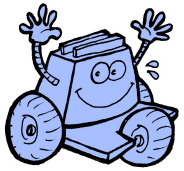
- Autonomy
  - Robots have to be capable of achieving task objectives without human input
  - Robots have to be able to make and execute their own decisions based on sensor information
- Intuitive Human-Robot Interfaces
  - Use of robots in smart homes can not require extensive user training
  - Commands to robots should be natural for inhabitants
- Adaptation
  - Robots have to be able to adjust to changes in the environment



Robots for Intelligent Environments

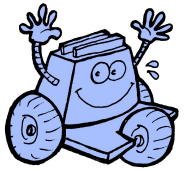
- Service Robots
  - Security guard
  - Delivery
  - Cleaning
  - Mowing
- Assistance Robots
  - Mobility
  - Services for elderly and People with disabilities

@HWU



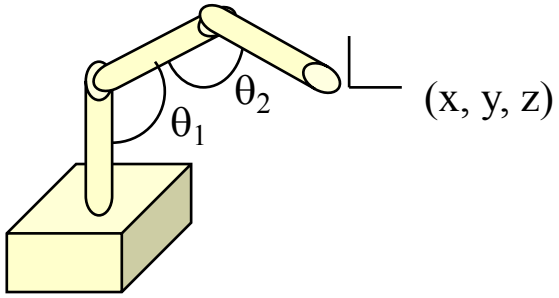
Autonomous Robot Control

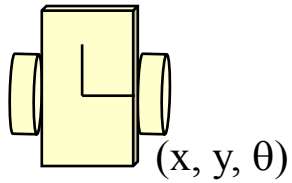
- To control robots to perform tasks autonomously a number of tasks have to be addressed:
  - Modeling of robot mechanisms
    - Kinematics, Dynamics
  - Robot sensor selection
    - Active and passive proximity sensors
  - Low-level control of actuators
    - Closed-loop control
  - Control architectures
    - Traditional planning architectures
    - Behavior-based control architectures
    - Hybrid architectures



Modeling the Robot Mechanism

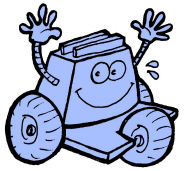
- Forward kinematics describes how the robots joint angle configurations translate to locations in the world





- Inverse kinematics computes the joint angle configuration necessary to reach a particular point in space.

- Jacobians calculate how the speed and configuration of the actuators translate into velocity of the robot



Mobile Robot Odometry

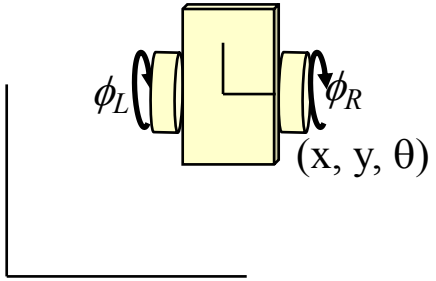
- In mobile robots the same configuration in terms of joint angles does not identify a unique location
  - To keep track of the robot it is necessary to incrementally update the location (this process is called odometry or dead reckoning)

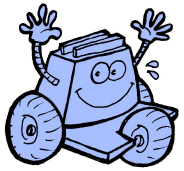
$$\begin{pmatrix} x \\ y \\ \theta \end{pmatrix}^{t+\Delta t} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix}^{t} + \begin{pmatrix} v_x \\ v_y \\ \varpi \end{pmatrix} \Delta t$$

  - Example: A differential drive robot

$$v_x = \cos(\theta)\frac{r(\dot{\phi}_L + \dot{\phi}_R)}{2}, v_y = \sin(\theta)\frac{r(\dot{\phi}_L + \dot{\phi}_R)}{2}$$

$$\varpi = \frac{r}{d}\left(\dot{\phi}_L - \dot{\phi}_R\right)$$

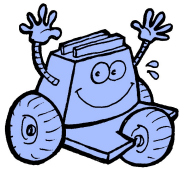




Actuator Control

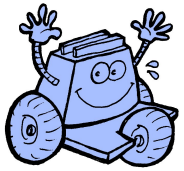
- To get a particular robot actuator to a particular location it is important to apply the correct amount of force or torque to it.
  - Requires knowledge of the dynamics of the robot
    - Mass, inertia, friction
    - For a simplistic mobile robot: $F = m\,a + B\,v$
  - Frequently actuators are treated as if they were independent (i.e. as if moving one joint would not affect any of the other joints).
  - The most common control approach is PD-control (proportional, differential control)
    - For the simplistic mobile robot moving in the x direction:

$$F = K_P\left(x_{desired} - x_{actual}\right) + K_D\left(v_{desired} - v_{actual}\right)$$



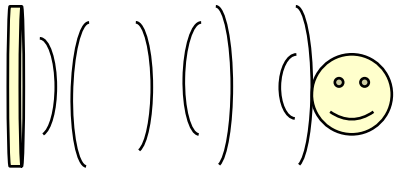
Robot Navigation

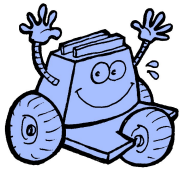
- Path planning addresses the task of computing a trajectory for the robot such that it reaches the desired goal without colliding with obstacles
  - Optimal paths are hard to compute in particular for robots that can not move in arbitrary directions (i.e. nonholonomic robots)
  - Shortest distance paths can be dangerous since they always graze obstacles
  - Paths for robot arms have to take into account the entire robot (not only the end effector)



Sensor-Driven Robot Control

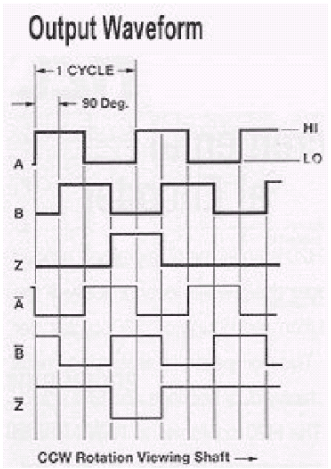
- To accurately achieve a task in an intelligent environment, a robot has to be able to react dynamically to changes ion its surrounding
  - Robots need sensors to perceive the environment
  - Most robots use a set of different sensors
    - Different sensors serve different purposes
  - Information from sensors has to be integrated into the control of the robot

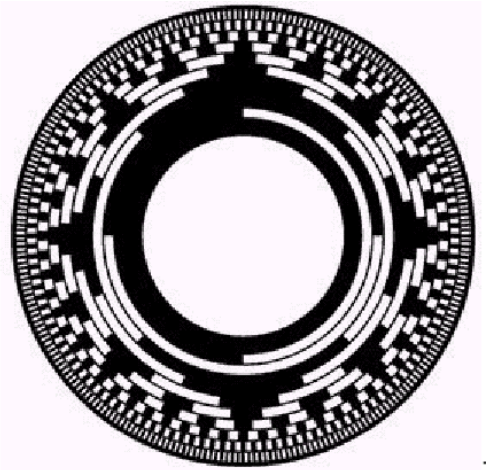




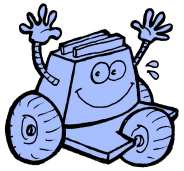
Robot Sensors

- Internal sensors to measure the robot configuration
  - Encoders measure the rotation angle of a joint



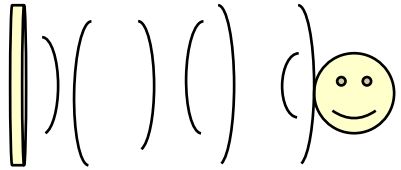


  - Limit switches detect when the joint has reached the limit

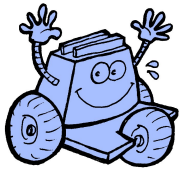


Robot Sensors

- Proximity sensors are used to measure the distance or location of objects in the environment. This can then be used to determine the location of the robot.
  - Infrared sensors determine the distance to an object by measuring the amount of infrared light the object reflects back to the robot
  - Ultrasonic sensors (sonars) measure the time that an ultrasonic signal takes until it returns to the robot

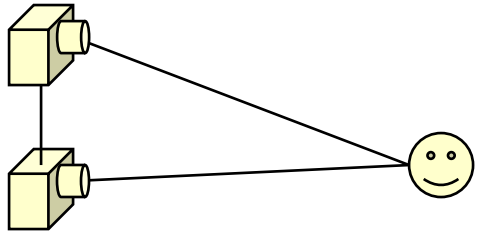


  - Laser range finders determine distance by measuring either the time it takes for a laser beam to be reflected back to the robot or by measuring where the laser hits the object

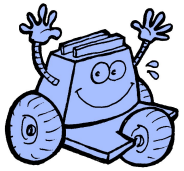


Robot Sensors

- Computer Vision provides robots with the capability to passively observe the environment
  - Stereo vision systems provide complete location information using triangulation

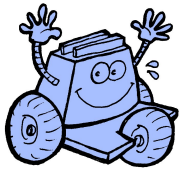


  - However, computer vision is very complex
    - Correspondence problem makes stereo vision even more difficult



Uncertainty in Robot Systems

- Robot systems in intelligent environments have to deal with sensor noise and uncertainty
  - Sensor uncertainty
    - Sensor readings are imprecise and unreliable
  - Non-observability
    - Various aspects of the environment can not be observed
    - The environment is initially unknown
  - Action uncertainty
    - Actions can fail
    - Actions have nondeterministic outcomes

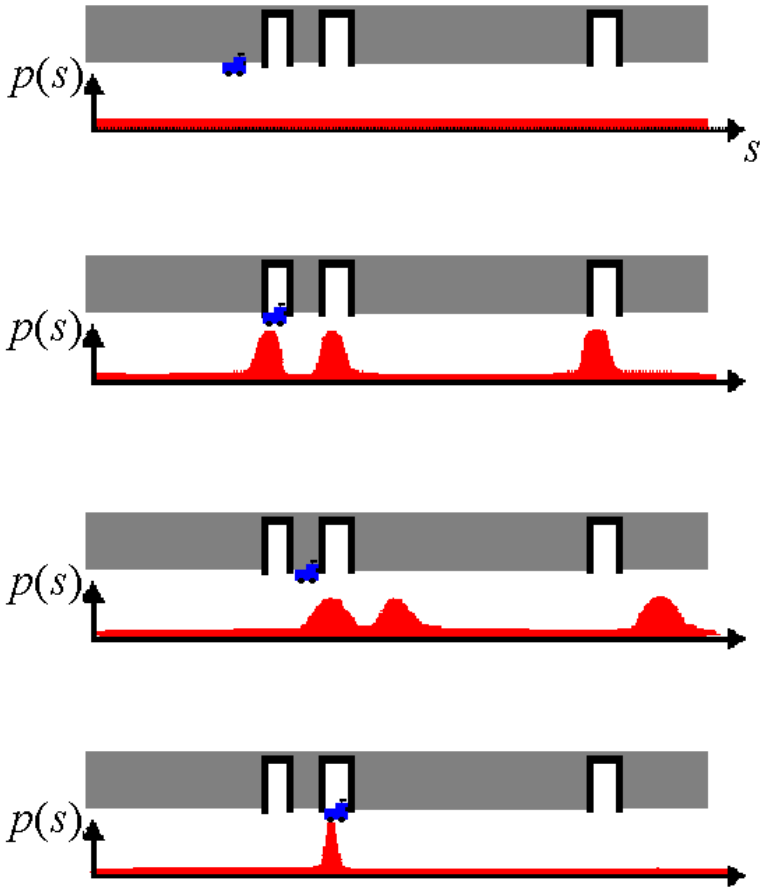


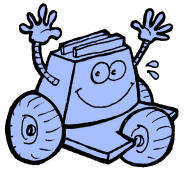
Probabilistic Robot Localization

- Explicit reasoning about Uncertainty using Bayes filters:

$$b(x_t) = \eta\ p(o_t \mid x_t) \int p(x_t \mid x_{t-1}, a_{t-1})\ b(x_{t-1})\ dx_{t-1}$$

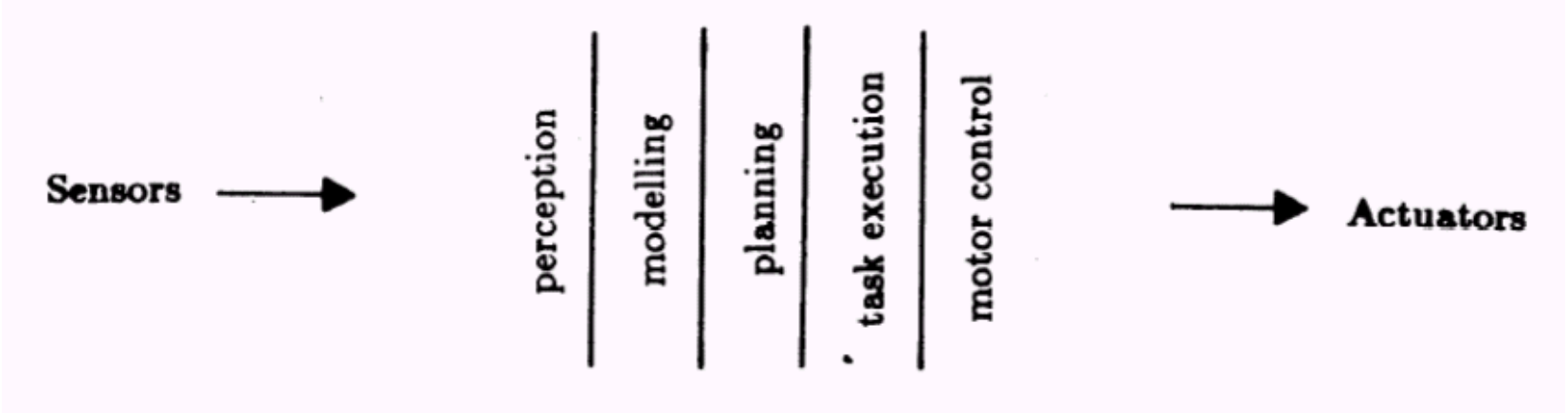
- Used for:
  - Localization
  - Mapping
  - Model building



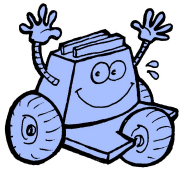


Deliberative Robot Control Architectures

- In a deliberative control architecture the robot first plans a solution for the task by reasoning about the outcome of its actions and then executes it

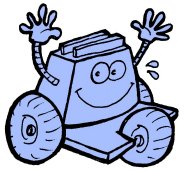


- Control process goes through a sequence of sensing, model update, and planning steps



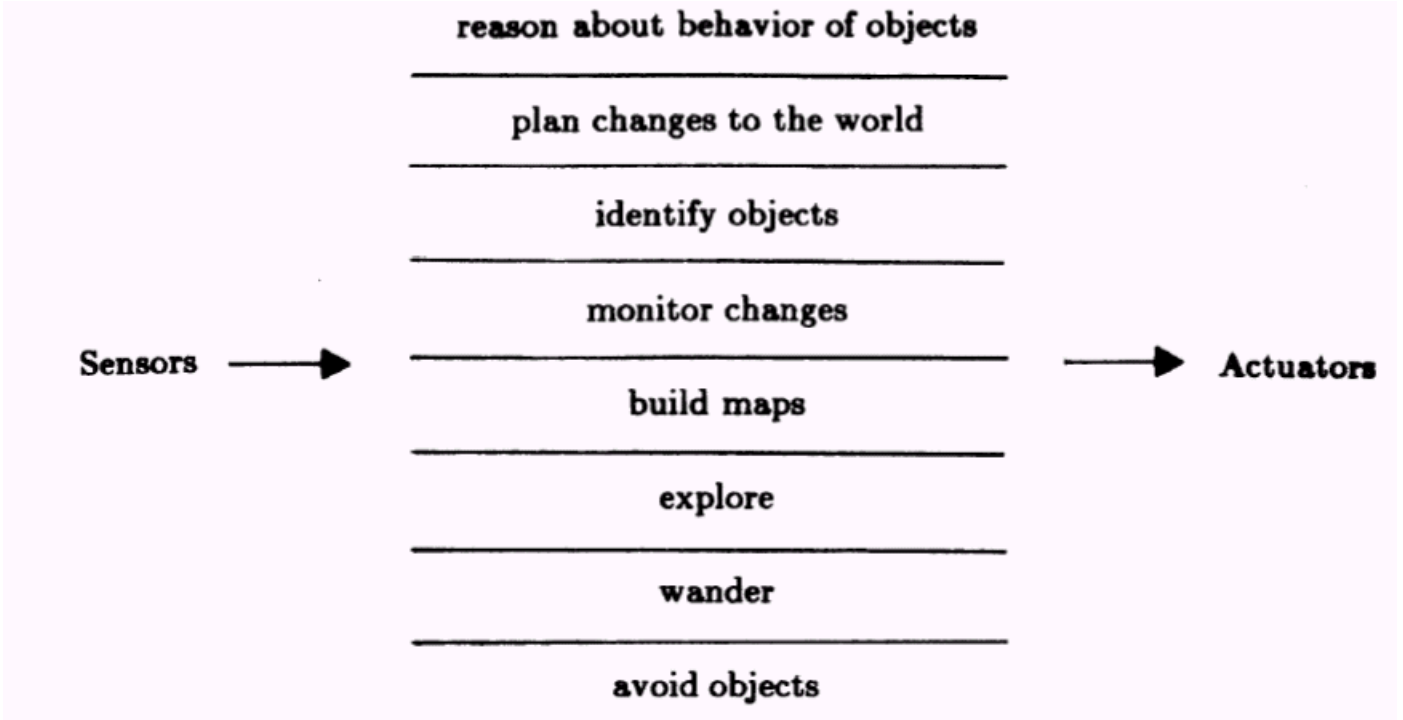
Deliberative Control Architectures

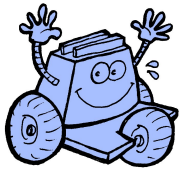
- Advantages
  - Reasons about contingencies
  - Computes solutions to the given task
  - Goal-directed strategies
- Problems
  - Solutions tend to be fragile in the presence of uncertainty
  - Requires frequent re-planning
  - Reacts relatively slowly to changes and unexpected occurrences



Behavior-Based Robot Control Architectures

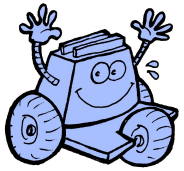
- In a behavior-based control architecture the robot's actions are determined by a set of parallel, reactive behaviors which map sensory input and state to actions.





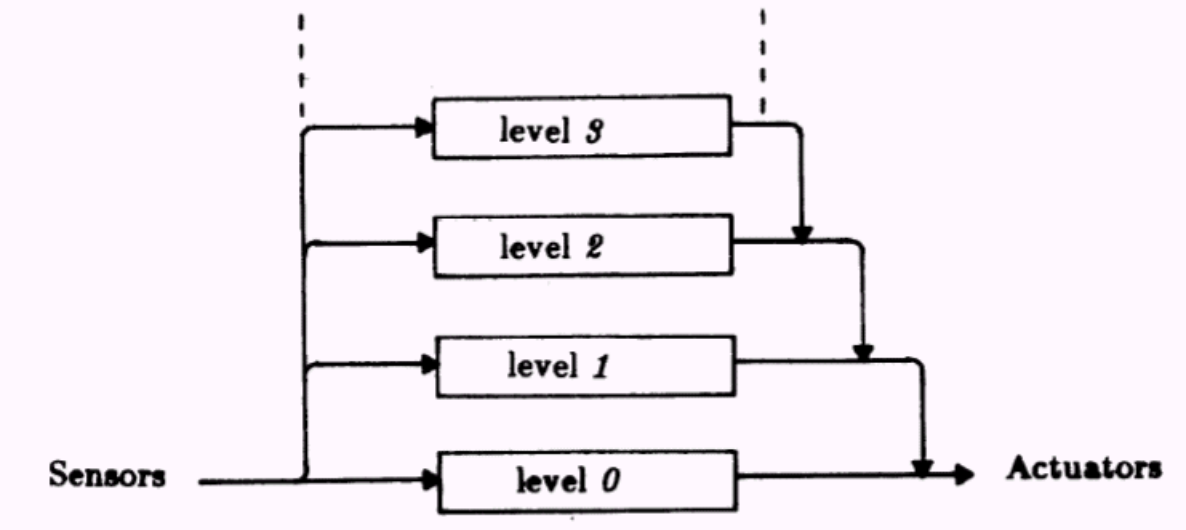
Behavior-Based Robot Control Architectures

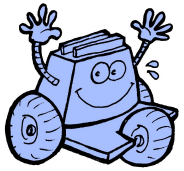
- Reactive, behavior-based control combines relatively simple behaviors, each of which achieves a particular subtask, to achieve the overall task.
  - Robot can react fast to changes
  - System does not depend on complete knowledge of the environment
  - Emergent behavior (resulting from combining initial behaviors) can make it difficult to predict exact behavior
  - Difficult to assure that the overall task is achieved



Behavior-Based Architectures: Subsumption Example

- Subsumption architecture is one of the earliest behavior-based architecture
  - Behaviors are arranged in a strict priority order where higher priority behaviors subsume lower priority ones as long as they are not inhibited.

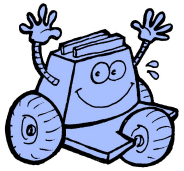




Reactive, Behavior-Based Control Architectures

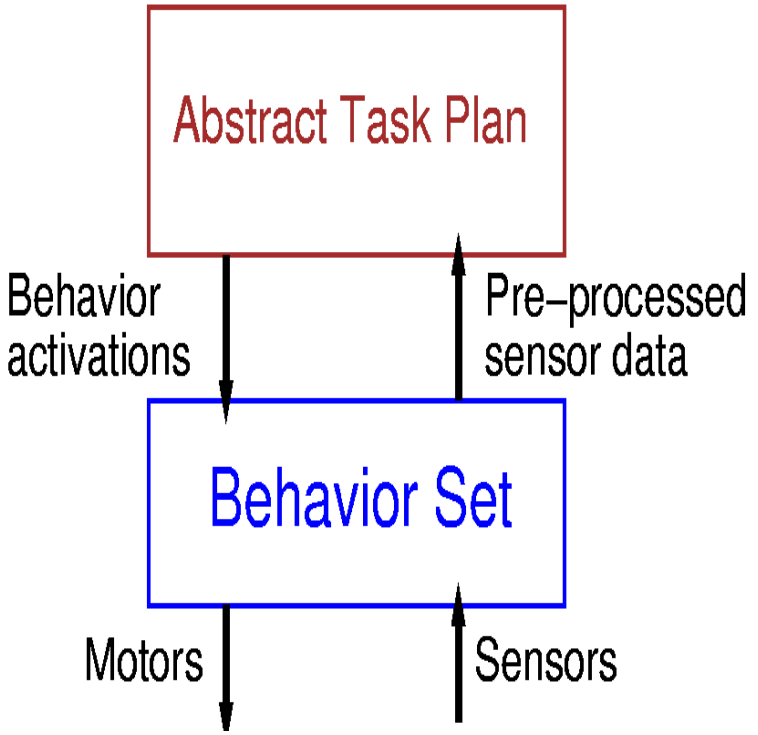
- Advantages
  - Reacts fast to changes
  - Does not rely on accurate models
    - “The world is its own best model”
  - No need for re-planning

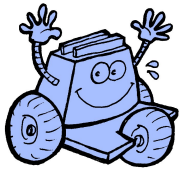
- Problems
  - Difficult to anticipate what effect combinations of behaviors will have
  - Difficult to construct strategies that will achieve complex, novel tasks
  - Requires redesign of control system for new tasks



Hybrid Control Architectures

- Hybrid architectures combine reactive control with abstract task planning
  - Abstract task planning layer
    - Deliberative decisions
    - Plans goal directed policies
  - Reactive behavior layer
    - Provides reactive actions
    - Handles sensors and actuators





Hybrid Control Architectures

- Advantages
  - Permits goal-based strategies
  - Ensures fast reactions to unexpected changes
  - Reduces complexity of planning
- Problems
  - Choice of behaviors limits range of possible tasks
  - Behavior interactions have to be well modeled to be able to form plans